Developments in Structural Learning Using Ihara Coefficients and Hypergraph Representations

Peng Ren

A Thesis Submitted for the Degree of Doctor of Philosophy

Departments of Computer Science
University of York
Deramore Lane
York Y010 5GH

December 2010

Abstract

This thesis addresses the problems in structural learning, particularly focusing on structural characterization and matching. To this end, we present an approach named *Ihara coefficients*, which is capable of characterizing structures of varying order into pattern space related to prime cycles. Furthermore, we develop a matching algorithm for establishing correspondences between structures by conducting *dominant cluster analysis* (DCA) on a *direct product hypergraph* (DPH) .

In Chapter 3 we describe how to extract characteristics from the Ihara zeta function for the purpose of clustering graphs. The novel contributions of this chapter are twofold. First, we demonstrate how to characterize unweighted graphs in a permutation invariant manner using the polynomial coefficients from the Ihara zeta function, i.e. the Ihara coefficients. Second, we generalize the definition of the Ihara coefficients to edge-weighted graphs. For an unweighted graph, the Ihara zeta function is the reciprocal of a quasi characteristic polynomial of the adjacency matrix of the associated oriented line graph. Since the Ihara zeta function has poles which give rise to infinities, the most convenient numerically stable representation is to work with the coefficients of the quasi characteristic polynomial. Moreover, the polynomial coefficients are invariant to vertex order permutations and also convey information concerning the cycle structure of the graph. To generalize the representation to edge-weighted graphs we make use of the reduced Bartholdi zeta function. We prove that the computation of the Ihara coefficients for unweighted graphs is a special case of our proposed method for unit edge-weights. We also present a spectral analysis of the Ihara coefficients and indicate their advantages over other graph spectral methods. We apply the proposed graph characterization method to capturing graph-class structure and clustering graphs. Experimental results reveal that the Ihara coefficients are more effective than methods based on Laplacian spectra.

In Chapter 4 we aim to seek a compact characterization of nonuniform unweighted hypergraphs for the purposes of clustering. To this end, we develop a polynomial characterization for hypergraphs based on the Ihara zeta function. We investigate the flexibility of the polynomial coefficients for learning relational structures with different relational orders. Furthermore, we develop an efficient method for computing the coefficient set. Our representation for hypergraphs takes into account not only the vertex connections but also the hyperedge cardinalities, and thus can distinguish different relational orders, which is prone to ambiguity if the hypergraph Laplacian is used. In our experimental evaluation, we demonstrate the effectiveness of the proposed characterization for clustering nonuniform unweighted hypergraphs and its advantages over the spectral characterization of the hypergraph Laplacian.

In addition to the flexible characterization algorithms developed based on the Ihara coefficients, we present a novel structural matching algorithm in Chapter 5. This algorithm can be used both for pairwise matching and higher order matching. We formulate the problem of high order structural matching by applying *dominant cluster analysis* (DCA) to a *direct product hypergraph* (DPH) (essentially the extension of the association graph idea to hypergraphs). For brevity we refer to the resulting algorithm as DPH-DCA. The starting point for our method is to construct a K-uniform direct product hypergraph for the two sets of higher-order features to be matched. Each vertex in the direct product hypergraph represents a potential correspondence and the weight on each hyperedge represents the agreement between two K-tuples drawn from the two feature sets. Vertices representing correct assignments tend to form a strongly intra-connected cluster, i.e. a dominant cluster. We evaluate the association of each vertex belonging to the dominant cluster by maximizing an objective function which maintains the K-tuple agreements. The potential correspondences with non-zero association weights are more likely to belong to the dominant cluster than the remaining zero-weighted ones. They are thus selected as correct matchings subject to the one-to-one correspondence constraint. Furthermore, we present a

route to improving the matching accuracy by invoking prior knowledge and show how the available outliers can be rejected to avoid contextual ambiguity. The effectiveness of the overall DPH-DCA framework can also be justified in terms of evolutionary game theory, and we hereby observe that the optimal solution obtained by DPH-DCA achieves a Nash equilibrium subject to the Karush-Kuhn-Tucker (KKT) conditions. An experimental evaluation shows that our method outperforms several state-of-the-art higher order structural matching methods both in terms of immunity to additive noise and robustness to outliers.

All the methods presented in this thesis will allow themselves to be applied to both pairwise and higher order relational patterns, and thus provide a route to addressing the structural characterization and matching problems by invoking multiple relationships.

Acknowledgments

First and foremost, I would like to express my sincere appreciation and gratitude to my supervisor, Prof. Edwin R. Hancock, for his support and advice on my research during the period of my PhD study. I have learnt a lot from Edwin, especially in how to formulate an academic problem in a theoretical way and how to present the research outcome in a professional manner. Also, I would like to thank Prof. Richard C. Wilson for his valuable suggestions and impartial assessment on my work, and Prof. Horst Bunke for his effort in examining the thesis. My thanks also go to Dr. Adrian Bors, Dr. Will Smith and all the friends at York for their help, kindness and friendship.

There are some other researchers who have provided valuable help on my research. In this respect, I thank Miss Tatjana Aleksić and Dr. Christopher K. Storm for their constructive suggestions on the work in this thesis. I also thank Dr. Samuel Rota Bulò for providing the code of the game-theoretic clustering algorithm and Dr. David Emms for providing the code for computing the quantum walk.

Especially, I want to thank Ms. Filomena M. Ottaway for all her help from the very beginning to the very end of my PhD study.

Last but certainly not least, I want to thank my family for their continuous support during the period of my PhD study.

Declaration

I hereby declare that all the work in this thesis is solely my own, except where attributed and cited to another author. Most of the material in this thesis has been previously published by the author. A complete list of publications can be found on page iii.

List of Publications

Journal Papers

1. Peng Ren, Richard C. Wilson and Edwin R. Hancock. Graph Characterization via Ihara Coefficients. *IEEE Transactions on Neural Networks*, 22(2):233-245, 2011.

2. Peng Ren, Tatjana Aleksić, Richard C. Wilson and Edwin R. Hancock. A Polynomial Characterization of Hypergraphs Using the Ihara Zeta Function. To appear in *Pattern Recognition*.

3. Peng Ren, Tatjana Aleksić, David Emms, Richard C. Wilson and Edwin R. Hancock. Quantum Walks, Ihara Zeta Functions and Cospectrality in Regular Graphs. To appear in *Quantum Information Processing*.

Conference Papers

1. Peng Ren, Tatjana Aleksić, Richard C. Wilson and Edwin R. Hancock. Ihara Coefficients: A Flexible Tool for Higher Order Learning. In *Proceedings of Joint IAPR International Workshop Structural, Syntactic, and Statistical Pattern Recognition*, 2010.

2. En Zhu, Edwin R. Hancock, Peng Ren, Jianping Yin and Jianming Zhang. Associating Minutiae between Distorted Fingerprints Using Minimal Spanning Tree. In

Proceedings of 7th International Conference on Image Analysis and Recognition, 2010.

3. Peng Ren, Richard C. Wilson and Edwin R. Hancock. Weighted Graph Characteristics from Oriented Line Graph Polynomials. In *Proceedings of 12th International Conference on Computer Vision*, 2009.

4. Peng Ren, Tatjana Aleksić, Richard C. Wilson and Edwin R. Hancock. Hypergraphs, Characteristic Polynomials and the Ihara Zeta Function. In *Proceedings of 13th International Conference on Computer Analysis of Images and Patterns*, 2009.

5. Peng Ren, Richard C. Wilson and Edwin R. Hancock. Characteristic Polynomial Analysis on Matrix Representations of Graphs. In *Proceedings of 7th IAPR-TC-15 International Workshop on Graph-Based Representations in Pattern Recognition*, 2009.

6. Peng Ren, Richard C. Wilson and Edwin R. Hancock. Pattern Vectors from the Ihara Zeta Function. In *Proceedings of 19th International Conference on Pattern Recognition*, 2008.

7. Shengping Xia, Peng Ren and Edwin R. Hancock. Ranking the Local Invariant Features for the Robust Visual Saliencies. In *Proceedings of 19th International Conference on Pattern Recognition*, 2008.

8. Peng Ren, Richard C. Wilson and Edwin R. Hancock. Graph Characteristics from the Ihara Zeta Function. In *Proceedings of Joint IAPR International Workshop Structural, Syntactic, and Statistical Pattern Recognition*, 2008.

9. Peng Ren, Richard C. Wilson and Edwin R. Hancock. Spectral Embedding of Feature Hypergraphs. In *Proceedings of Joint IAPR International Workshop Structural, Syntactic, and Statistical Pattern Recognition*, 2008.

Contents

List of Figures

List of Tables

Glossary of Notation

$G(V, E)$	Graph with vertex set V and edge set E
$H(V, E_H)$	Hypergraph with vertex set V and edge set E_H
$\boldsymbol{A}$	Adjacency matrix
$\boldsymbol{L}$	Laplacian matrix
$\boldsymbol{H}$	Incidence matrix
$\boldsymbol{T}$	Perron-Frobenius operator
$Z_G(\cdot)$	Ihara zeta function for graphs
$\zeta_H(\cdot)$	Ihara zeta function for hypergraphs
c_i	The ith Ihara coefficient
λ_i	The ith smallest eigenvalue of the Laplacian matrix $\boldsymbol{L}$
S_b	Between class scatter
S_w	Within class scatter
J_C	Criterion function for discrimination
$\mathcal{C}$	Compatibility tensor
$\mathcal{H}$	Adjacency tensor
$\Pr(\cdot)$	Probability function
$i \leftrightarrow i'$	Correspondence between vertices i and i'
$\mathbf{P}$	Matching matrix
$\mathbf{p}$	Association probability vector
$S(\cdot)$	Function of similarity measure

Chapter 1

Introduction

In this chapter we provide an introduction and motivation for the research work presented in this thesis, explaining why we are interested in structural characterization and matching. We commence by introducing the problems encountered in learning with structured data. Then we briefly describe the possible alternative approaches to these problems, followed by our research goals and contributions. Finally, an outline of the thesis is provided at the end of this chapter.

1.1 The Problems

Structured data have been widely used for representing relational patterns. Pattern analysis tasks involving trees, graphs and hypergraphs arise in a number of domains such as natural language processing, proteomics/chemoinformatics, data mining, computer vision and complex systems.

In computer vision graph-structures are widely used to abstract image structure. Furthermore, to capture the multiple relationships between visual features, hypergraphs have been exploited as a more sophisticated representation for image abstraction. The first steps in extracting (hyper)graph structures from images are those of segmentation and

perceptual grouping. However, the algorithms used to segment the image primitives are not reliable. As a result there are both additional and missing vertices in these extracted graphs due to segmentation errors and variations arise accordingly. Thus, structural learning methods, which are capable of capturing the variations both between and within different classes of structures, are needed.

However, learning with (hyper)graphs is difficult. One important reason is that (hyper)graphs are not vectors and hence can not be easily summarized. This renders the problem of characterizing the mean and covariance of structure categories in their original forms intractable. Moreover, there is no natural ordering of vertices and (hyper)edges and special algorithms are needed to establish correspondences. Due to these difficulties, relatively little methodology is available straightforwardly for learning with (hyper)graph structures. In particular, vectorial methods from statistical machine learning can not be easily applied to the structured data, since there is no canonical ordering of the vertices in a (hyper)graph.

There are several approaches to coping with the difficulties that arise in learning with structural patterns. One possible route is to work with permutation invariant graph characteristics that relate to the topological structure, such as vertex number, edge number, and (hyper)graph radius and perimeter. Alternative features that can be used to establish pattern vectors include (hyper)graph spectra and (hyper)graph polynomial characteristics. By characterizing (hyper)graphs using these features, we can embed (hyper)graphs in a vector space and manipulate them as point patterns. We can characterize structural variations in terms of statistical variations across the point patterns. Furthermore, we can measure (dis)similarities between a pair of (hyper)graphs by computing the feature distance between vectorial representations for the (hyper)graphs, and then perform pairwise clustering or embed sets of (hyper)graphs in a vector space using multi-dimensional scaling on the (dis)similarities.

Another possible method for overcoming the difficulties in (hyper)graph based learn-

ing is to establish a mapping between two vertex sets. This family of methods takes advantage of the likelihood that particular vertices and (hyper)edges co-occur and establishes correspondences between vertices of two structures. The matching strategy can be applied to learning modes of structural variation within a (hyper)graph class. Furthermore, the matching methods also play an important role in constructing generative models for graphs, because vertex correspondences are required as a prerequisite in learning the model and the model inference.

Structural characterization and matching are both approaches to addressing the difficulties in learning with structured data. However, the most exiting methods are confined to pairwise graphs and do not lend themselves to higher order relationship patterns, i.e. hypergraphs. Therefore, characterization and matching methods available for higher order relationships have been a longstanding quest in structural pattern recognition.

1.2 Our Goals

The overall goal of this thesis is to develop novel methods addressing the problems encountered in structural learning. Specifically,

i) We aim to establish a novel vectorial representation for graphs to which statistical learning methods can be applied. In particular, we will explore the Ihara zeta function, which is governed by the cycle frequencies of a graph. Based on the Ihara zeta function we will develop novel graph characterization methods incorporating topological properties, spectral features and polynomial characteristics of graphs. Furthermore, we will develop a method to extend the representation from unweighted graphs to edge-weighted graphs.

ii) We aim to make a polynomial analysis of hypergraphs by using the Ihara zeta function, to establish a novel hypergraph characterization method. We will show how to represent a hypergraph by using a colored oriented line graph in the Ihara zeta function. We will demonstrate the effectiveness of the new representation in avoiding order ambigui-

ties of the hypergraph Laplacian. Additionally, we will develop an efficient computational method that renders the computation of the new representation tractable in practice.

iii) We aim to develop a novel method for high order structural matching. Our strategy is to cast the high order matching problem into that of high order clustering. We will show how to establish correct correspondences by clustering vertices in a hypergraph constructed on the two sets of features with high order relationships to be matched. Additionally, we will provide a route to making use of prior knowledge of outliers to improve the matching performance.

1.3 Contributions

To achieve the research goals described in Section 1.2, we make the following specific contributions:

1.3.1 Graph Characterization Based on the Ihara Coefficients

We develop a new framework for characterizing graphs using features based on the Ihara zeta function. In its original form the Ihara zeta function is defined over the prime cycles of a graph and has poles corresponding to the lengths of the prime cycles. Hence, if we attempt to sample the Ihara zeta function to obtain a characterization, we will encounter numerical instabilities due to the associated infinities. However, since the Ihara zeta function is determined by the cycle frequencies in a graph, there should be a number of alternative numerically stable representations upon which we can draw. The Ihara zeta function is the reciprocal of the quasi characteristic polynomial of an oriented line graph. We therefore make use of the coefficients of the polynomial, i.e. the Ihara coefficients, as our features. The Ihara coefficients are determined by the cycle frequencies in the graph, and not only avoid the danger of infinities, but are also permutation invariants.

Unfortunately, the Ihara zeta function is only defined for unweighted graphs and can

not be applied to edge-weighted graphs. The reasons for this are twofold: first, when we encounter weighted edges, the cycle lengths are determined not only by the numbers of connected edges, but also by their cumulative weights; second, the determinant expression for the Ihara zeta function is determined by integer prime cycle lengths, and can not be easily generalized to weighted path length. To overcome these problems, we develop a novel method for computing the Ihara coefficients for edge-weighted graphs. This is effected by generalizing the determinant form of the Ihara zeta function using a reduced Bartholdi zeta function. We demonstrate that the computation of the Ihara coefficients for unweighted graphs is a special case of our proposed method for graphs of unit edge-weight. In this way we can accommodate both unweighted and edge-weighted graphs, and make the proposed approach a flexible method for both cases.

1.3.2 Polynomial Analysis of Hypergraphs Based on the Ihara Coefficients

We propose to use the Ihara zeta function for hypergraph characterization. To demonstrate the effectiveness of the Ihara zeta function, we first explain the shortcomings of the spectral methods for representing nonuniform unweighted hypergraphs theoretically and then present our proposed method to address it. Unlike the method introduced in [1] and those reviewed in [2], we do not establish graph approximations based on the original vertex set of a hypergraph. Instead we transform the hypergraph into a colored oriented line graph, in which the vertex set includes information concerning the relational orders of the original hypergraph vertices. This avoids attaching a weight to the hyperedge when it is unnecessary. Our characterization is based on the determinant form of the Ihara zeta function. In this way, we provide a matrix representation that naturally avoids the order ambiguity which might occur when the hypergraph Laplacian is used. Associated with the determinant is a set of characteristic polynomial coefficients which are referred to as the Ihara coefficients and constitute our representation for the hypergraph.

It is the discriminative ability that readily makes the Ihara coefficients a flexible method for distinguishing high order structures. Furthermore, we develop an efficient method for computing the hypergraph Ihara coefficients, which renders the computation of the coefficients tractable. We apply the proposed characterization method to clustering hypergraphs extracted from images of different object views and demonstrate their superiority to the hypergraph Laplacian and the normalized hypergraph Laplacian.

1.3.3 High Order Structural Matching Based on DPH-DCA

We develop a novel framework for high order matching based on *dominant cluster analysis* (DCA) on a *direct product hypergraph* (DPH). For brevity we refer to the framework as DPH-DCA. The idea is motivated by the concept of main cluster for graph matching [50] and its generalization for higher order matching [28]. However, we use a different algorithm to extract the dominant cluster, which not only outperforms the state-of-the-art methods but also satisfies basic probabilistic properties. Furthermore, we present a method for incorporating prior knowledge regarding outliers into our framework using a specific initialization. This improves the matching performance of our method and comparable results can not be achieved by using alternative high order matching algorithms [28][110]. We also justify the effectiveness of the DPH-DCA framework in terms of evolutionary game theory. The theoretical contribution here is the novel framework for high order matching in terms of clustering and its game theoretic interpretation. Furthermore, the proposed method also provides a route for a comprehensive understanding of the relationship between high order cluster and matching. The practical contribution is that our method outperforms the state-of-the-art high order matching methods in the practical experiments.

1.4 Thesis Outline

The rest of the thesis is organized as follows: Chapter 2 reviews the research literature on structural characterization and matching; Chapter 3 presents a graph characterization method based on the Ihara coefficients; Chapter 4 introduces a polynomial analysis of hypergraphs based on the Ihara coefficients; Chapter 5 describes a hypergraph matching method based on DPH-DCA; Chapter 6 concludes the work in this thesis and points out possible directions for future research. Additionally, Appendix A presents the relationship between the discrete-time quantum walks and the Ihara zeta function, which provides a new perspective on the research of the Ihara zeta function.

Chapter 2

Literature Review

Graphs and hypergraphs are important representations for structured data in pattern recognition. One aim of this thesis is to develop learning methods that can effectively characterize or match high order structures. In the light of this aim, we commence in Section 2.1 by reviewing fundamental graph characteristics along with zeta functions as a more sophisticated graph representation. Additionally, in this section we explain our motivation for conducting structural characterization using the Ihara zeta function. We then review the graph-based learning methods in Section 2.2, followed by an overview of hypergraph representation for structural pattern recognition in Section 2.3. Finally, we review the research literature on the structural matching in Section 2.4, in which we also describe the shortcomings of existing methods which motivate our novel framework for high order matching.

2.1 Graph Characteristics and Zeta Functions

Various statistical methods are available for learning patterns represented by vectors. However, these statistical methods are not suitable for structured data such as trees, graphs and hypergraphs. This is because structural patterns can not be easily converted into vec-

tors, and the difficulties arise in several aspects. First, there is no natural ordering for the vertices in an unlabeled structure, and this is in contrast to vector components that have a natural order. Second, the variation within a particular graph class may result in subtle changes in structures of individual graphs. This may involve different vertex set and edge set cardinalities for graphs drawn from the same class. Moreover, subspaces (e.g. eigenspaces) spanned by the matrix representations of graphs with different vertex set cardinalities are of different dimensions, and thus pattern vectors residing in the resulting subspaces would be of different lengths. All these difficulties need to be addressed if we want to apply the existing statistical methods to learning with structural patterns.

The task of structural characterization is to characterize classes of structural patterns into a feature space where statistical learning methods can be readily applied. To this end, the key issue is to extract from structural patterns a set of characteristics which not only exactly describe the individual structures but also capture the variations between/within the structure classes. In this regard, the most straightforward characteristics for graphs are the topological properties, such as vertex set cardinality, edge density, graph perimeter and volume [58]. Furthermore, by measuring the topological difference between graphs, graph edit distance can be neatly defined [76]. Bunke *et al.* [68][69][32] embed graphs into a feature space by using kernel strategies which adopt edit distance as a similarity measure. Within such graph characterization frameworks, graphs can be easily classified by using statistical learning approaches such as SVM. Although the topological features have a straightforward meaning concerning the structures, they are hard to enumerate for objects with a considerable size. The computational complexity for edit distance is exponential to the cardinality of the vertex set and is usually computationally prohibitive in practice, unless approximations are made subject to certain constraints [68]. These shortcomings limit the direct use of topological properties for the purpose of structural characterization.

Another approach to graph characterization is to extract alternative vertex permuta-

tion invariant characteristics straightforwardly from the matrix representations of graphs. Here the initial matrix representation $\boldsymbol{M}$ can be based either on the adjacency matrix, the Laplacian matrix or the signless Laplacian [26]. The definition of the adjacency matrix $\boldsymbol{A}$ for a graph $G(V, E)$ is as follows

$$A_{uv} = \begin{cases} w(u,v) & \text{if } \{u,v\} \in E; \\ 0 & \text{otherwise;} \end{cases} \tag{2.1}$$

where $w(u, v)$ is the weight attached to the edge $\{u, v\}$. For an unweighted graph, $w(u, v)$ is 1 if there is an edge between vertices u and v. The degree of a vertex $u \in V$, denoted by $d(u)$, is defined as

$$d(u) = \sum_{v:\{v,u\}\in E} w(u,v). \tag{2.2}$$

For an unweighted graph, the degree of a vertex is simply the number of vertices adjacent to it. For a graph $G(V, E)$ with $|V| = N$, the matrix $\boldsymbol{D} = \text{diag}(d(v_1), d(v_2), \ldots, d(v_N))$, with the vertex degrees on the diagonal and zeros elsewhere is referred to as the degree matrix.

The Laplacian matrix $\boldsymbol{L}$ of a graph $G(V, E)$ is defined as $\boldsymbol{L} = \boldsymbol{A} - \boldsymbol{D}$, with entries

$$L_{uv} = \begin{cases} W(u,v) & \text{if } \{u,v\} \in E; \\ -d(u) & \text{if } u = v; \\ 0 & \text{otherwise.} \end{cases} \tag{2.3}$$

The matrix representation can be characterized using its eigenvalues $\text{sp}(\boldsymbol{M})$ and eigenvectors (i.e. using spectral graph theory). For instance, Luo, Wilson and Hancock [58] have made use of graph spectra to construct a set of handcrafted permutation invariant spectral features for the purpose of clustering graphs. Kondor *et al.* [47] have presented an approach to extracting the skew spectrum from the adjacency matrix of a graph up to a combinatorial transformation, and incorporated it into SVM kernels for the classification of chemical molecules. Furthermore, the same authors have refined their spectral

method by considering the number as well as the position of labeled subgraphs in a given graph [48]. Though the spectral features appear to be less related to graph topology than the straightforward topological characteristics, the Laplacian spectra give a competitive performance in clustering graphs over various alternative methods [103].

For the graph matrix representation $\boldsymbol{M}$, the coefficients of its characteristic polynomial $\det(\lambda \boldsymbol{I} - \boldsymbol{M})$ can also be taken as graph characteristics. These coefficients are closely related to the eigenvalues of $\boldsymbol{M}$, i.e. the graph spectrum. Brooks [12] has generalized the computation of the coefficients of the characteristic polynomial using three different methods. His first method is to express the coefficients in terms of the eigenvalues of the matrix representation, his second method uses the relationship between the coefficients and the kth derivative of the associated determinant, and the third method is a brute force method using matrix elements. Thus, it is clear that the eigenvalue-based and polynomial-based approaches are closely related to each other and can lead to a number of practical graph characterizations. In this regard, pioneering research can be found in Wilson, Hancock and Luo's work [102] which shows how to extract a rich family of permutation invariants from a graph by applying elementary symmetric polynomials to the elements of the spectral matrix derived from the Laplacian matrix .

An alternative possible characterization method that has received relatively little attention in the computer vision and pattern recognition community is provided by the zeta functions. In number theory, the Riemann zeta function is determined by the locations of the prime numbers. Bai, Wilson and Hancock [4] have explored the use of a modified version of the Riemann zeta function as a means of characterizing the shape of the heat kernel trace of a graph. They have also shown that the derivative of the zeta function at the origin is related to the determinant of the Laplacian matrix. Another natural extension of the Riemann zeta function from prime numbers to graphs is the Ihara zeta function. The Ihara zeta function is determined by the set of prime cycles on a graph, and is detailed in [44] and [45]. Hashimoto [40] subsequently deduced explicit factorizations for

bi-regular bipartite graphs. Bass [7] has generalized Hashimoto's factorization to all finite graphs. Stark and Terras [88][89][90] have published a series of articles on the topic. They commence by presenting a survey of the Ihara zeta function and its properties. Their novel contribution is to generalize the Ihara zeta function to develop edge and path based variants. Recently, Storm has further developed and refined the Ihara zeta function for hypergraphs [91].

The Ihara zeta function draws on the reciprocal of a polynomial associated with a graph and is hence akin to methods from algebraic graph theory. However, it also relies upon a graph transformation. This is an interesting observation since the quest for improved alternatives to the adjacency and Laplacian matrices has been a longstanding quest in spectral graph theory. Recently, the signless Laplacian (i.e. the degree matrix plus the adjacency matrix) has been suggested. However, Emms *et al.* [31] have recently shown that a unitary matrix characterization of the oriented line graph can be used to reduce or even completely lift the cospectrality of certain classes of graph, including trees and strongly regular graphs. This points to the fact that one potentially profitable route to improving methods from spectral graph theory may reside in graph transformation.

Although the Ihara zeta function have been widely investigated in the mathematics literature, it has received little attention as a means of characterizing graphs in machine learning. Furthermore, to be rendered tractable for real world problems in pattern recognition, the issue of how to generate stable pattern vectors from the Ihara zeta function must be addressed. Zhao *et al.* [111] have recently used Savchenko's formulation of the zeta function [78], expressed in terms of cycles, to generate merge weights for clustering over a graph-based representation of pairwise similarity data. Their formulation is based on a representation of oriented line graphs, which is an intermediate step in the development of the Ihara zeta function studied in this work. Watanabe *et al.* [99] have presented an approach to the analysis of Loopy Belief Propagation (LBP) by establishing a formula that connects the Hessian of the Bethe free energy with the edge Ihara zeta function.

A substantial part of this thesis is concerned with the Ihara zeta function. The motivation for us to explore the Ihara zeta function for structural characterization is twofold: first, the Ihara zeta function is determined by cycle frequencies, and thus capable of reflecting graph topologies; second, the Ihara zeta function can be expressed in a polynomial form of a transformed graph such that certain polynomial and spectral analysis can be done based on it. These properties of the Ihara zeta function allow it to naturally incorporate topologies, spectra and polynomials into a unifying representation, and thus enable it to have the potential to result in a rich family of structural characteristics.

2.2 Graph Representations for Pattern Recognition

This section reviews the various graph representations used in pattern recognition, not restricted to graph characterization. Graph-based methods are widely used in solving problems in computer vision and pattern recognition at different levels of feature abstraction. Early work related to graph-based representations focuses on identifying subgraph isomorphism [94] or measuring edit distance [76] for the purpose of structural pattern recognition. These methods enumerate the node attributes to obtain an optimal solution to certain cost functions. Therefore, graphs are not characterized in a mathematically consistent way by using these methods. However, this shortcoming can be overcome by adopting graph spectral methods [22] for graph characterization. In addition to representing graphs in terms of vertex set and edge set, another graph representation used in spectral graph theory is adjacency matrix or Laplacian matrix. Each entry of the matrix is associated with the pairwise relationship between two vertices, and the indices of the entry represent labels for the two vertices. By using the matrix representations, graphs can be processed in a computationally efficient and consistent way, because existing computing algorithms for matrices can be straightforwardly applied to graphs. Therefore, many statistical pattern recognition algorithms can directly work on graph-based data once the

matrix representations are established. One good example is to formulate the problem of clustering as that of computing the principal eigenvector of the normalized affinity matrix for a graph [100]. Furthermore, Zass *et al.* [109] have shown how to provide a probabilistic interpretation for this formulation by developing a completely positive factorization scheme. On the other hand, Shi *et al.* have [87] presented a method based on the normalized Laplacian matrix rather than the normalized affinity matrix. Their method is referred to as normalized cut because it is capable of balancing the cut and the association. Robles-Kelly *et al.* [70] have introduced a probabilistic framework based on a Bernoulli model which adopts EM algorithm for extracting the leading eigenvector as the cluster membership indicators. Pavan *et al.* [64] have formulated the problem of pairwise clustering as that of extracting the dominant set of vertices from a graph. Based on this notation, Rota-Bulo *et al.* have developed game-theoretic approaches to partial clique enumeration [74] and hypergraph clustering [73]. Qiu *et al.* [65] have characterized the random walk on a graph using the commute time between vertices and proved that the commute time matrix is a more robust measure of the proximity of data than the raw proximity matrix. Behmo *et al.* [9] have exploited the formulation based on commute times as a manner of image representation. Furthermore, some researchers have investigated the problem of graph based learning by incorporating the path-based information between vertices as a replacement of pairwise similarity. Representative work includes Path-Based Clustering [33] and the sum-over-paths covariance kernel [59].

Different from clustering graph vertices, the research on graph embedding aims to seek a low dimensional coordinates for the vertices. This is often conducted in a manifold learning scenario, where certain local features of the manifold underlying the original data are preserved. Based on a similar notion to normalized cut, Belkin *et al.* [8] have presented a graph embedding framework called Laplacian eigenmaps for dimensionality reduction. Other notable manifold learning methods include ISOMAP [92] and LLE [75]. These manifold learning methods adopt different cost functions and thus result in differ-

ent local structure preservations. Recently, Yan *et al.* [105] have generalized traditional embedding methods such as PCA by using a graph embedding framework and extended it into non-negative versions [107][98][55]. Shaw *et al.* [85] have introduced an embedding strategy which preserves the global topological properties of an input graph.

A preliminary step for all these graph-based methods (both for clustering and embedding) is to establish a graph over the training data. Data samples are represented as vertices of the graph and the edges represent the pairwise relationships between them. The methods for establishing a graph and measuring vertex similarities (i.e. edge weights) have a great influence on the subsequent graph-based learning algorithms. Therefore, the process of graph construction has recently attracted much research interest [60][27][46] as it remains only partially solved.

In addition to representing the pairwise relationship within a training data set (i.e. normalized cut, ISOMAP and LLE), graph-based methods also play an important role in learning with structured data. Problems of this kind arise when training data are not represented in vectors but in terms of relational structures such as trees and graphs. In this case, learning algorithms which admit structured data are needed. For example, the problem of discovering shape and object categories is frequently posed as one of clustering a set of graphs. This is an important process since it can be used to organize large databases of graphs in a manner that renders retrieval efficiency [80].

The strategies for learning with structured data can be roughly classified into two categories. The first is the graph characterization methods reviewed in Section 2.1. The second is to develop specific learning algorithms which admit individual graphs or trees as input. For the second category, a similarity between structured data samples is defined and traditional learning schemes are applied based on the pairwise similarities between structured data samples. For example, graph similarities can be computed by using tree or graph edit distance and structured objects are assigned to classes using pairwise clustering [15][93]. However, the use of pairwise similarities alone is a rather crude way to capture

the modes of variation present in graphs of a particular class. Moreover, it requires the computation of vertex correspondences which is sometimes an unreliable process. The graph kernels [66][97] overcome this problem to a certain degree by naturally incorporating vertex correspondences into the process of learning. This is effected by the learning process in which every pair of vertices drawn separately from two graphs are compared to obtain an entry of the kernel matrix. However, the process of vertex enumeration gives rise to computational inefficiency. Although fast computational scheme [86] has recently been proposed, these methods still undergo heavy computational overheads. In contrast to the graph kernel strategies, graph characterization methods would be more efficient if pattern vectors are suitably established, because it avoids enumerating the comparisons between every pair of vertices.

2.3 Hypergraph Representations for Pattern Recognition

There has recently been an increasing interest in hypergraph-based methods for representing and processing structures where the relations present are not simply pairwise. The main reason for this trend is that hypergraph representations allow vertices to be multiply connected by hyperedges and can hence capture multiple relationships between features. Due to their effectiveness in representing multiple relationships, hypergraph-based methods have been applied to various practical problems such as partitioning netlists [38] and clustering categorial data [34]. For visual processing, to the best of our knowledge, the first attempt at representing visual objects using hypergraphs dates back to Wong *et al.*'s [104] framework for 3D object recognition. In this work a 3D object model based on a hypergraph representation is constructed, and this encodes the geometric and shape information with polyhedrons as vertices and hyperedges. Object synthesis and recognition tasks are performed by merging and partitioning the vertex and hyperedge set. The method is realized using set operations and the hypergraphs are not characterized in a

mathematically consistent way. Later, Bretto *et al.* [11] introduced a hypergraph model for image representation, where they successfully and simultaneously solved the problems of image segmentation, noise reduction and edge detection. However, their method also relies on a crude form of set manipulation. Agarwal *et al.* [1] have performed visual clustering by partitioning a weighted graph transformed from the original hypergraph by a weighted sum of its hyperedges into the graph edge. Recently, Rota-Bulo *et al.* [72] have established a hypergraph model for estimating affine parameters in vision problems. Bunke *et al.* [14] have developed a hypergraph matching algorithm for object recognition, where consistency checks are conducted on hyperedges. The computational paradigm underlying their method is based on tree search operations. Zass *et al.*[110] and Duchenne *et al.*[28] have separately applied high-degree affinity arrays (i.e. tensors) to formulating hypergraph matching problems up to different cost functions. Both methods address the matching process in an algebraic manner but must undergo intractable computational overheads if hyperedges are not suitably sampled. Shashua *et al.* [83][84] have performed visual clustering by adopting tensors for representing uniform hypergraphs (i.e. those for which the hyperedges have identical cardinality) extracted from images and videos. Their work have been complemented by He *et al.*'s [43] algorithm for detecting number of clusters in tensor-based framework. Similar methods include those described in [36][18][19][20][73], in which tensors (uniform hypergraphs) have been used to represent the multiple relationships between objects. Additionally, tensors have recently been used to generalize dimensionality reduction methods based on linear subspace analysis into higher orders [96][106][42][41]. However, the tensor representation considers all possible permutations of a subset of vertices and establishes hyperedges with cardinality consistent with the relational order. Therefore, tensors can only represent uniform hypergraphs, and are not suited for nonuniform hypergraphs (i.e. hypergraphs with varying hyperedge cardinalities).

One common feature of these existing hypergraph representations is that they exploit

domain specific and goal directed representations. Specifically, most of them are confined to uniform hypergraphs and do not lend themselves to generalization. The reason for this lies in the difficulty in formulating a nonuniform hypergraph in a mathematically neat way for computation. There has yet to be a widely accepted and consistent way for representing and characterizing nonuniform hypergraphs, and this remains an open problem when exploiting hypergraphs for machine learning. Moreover, to be easily manipulated, hypergraphs must be represented in a mathematically consistent form, using structures such as matrices or vectors.

Since Chung's [23] definition of the Laplacian matrix for K-uniform hypergraphs, there have been several attempts to develop matrix representations of hypergraphs. To establish the adjacency matrix and Laplacian matrix for a hypergraph, an equivalent graph representation is often required. Once the graph approximation is to hand, its graph representation matrices (e.g. the adjacency matrix (2.1) and the Laplacian matrix (2.3)) are referred to as the corresponding hypergraph representation matrices. It is based on these approximate matrix representations that the subsequential processes of hypergraphs (e.g. high order clustering and matching) take place. Agarwal *et al.* [2] have compared a number of alternative graph representations [10][34][54][71][112] for hypergraphs and explained their relationships with each other in machine learning. One common feature for these methods, as well as the method in [1], is that a weight is assumed to be associated with each hyperedge. Additionally, the graph representations for a hypergraph can be classified into two categories: (a) the clique expansion [1][10][34][71] and (b) the star expansion [54][112]. The clique expansion represents a hypergraph by constructing a graph with all pairs of vertices within a hyperedge connecting to each other. The star expansion represents a hypergraph by introducing a new vertex to every hyperedge, and constructing a graph with all vertices within a hyperedge connecting to the newly introduced vertex. In both strategies, each edge in each individual graph representation is weighted in a manner determined by the corresponding hyperedge weight in a task-specific way that is differ-

ent from others. Moreover, these graph-based representations for hypergraphs are just approximations and give rise to information loss, as reported in [1]. This deficiency may result in ambiguities when the approximation methods are used to distinguish structures with different relational orders.

To address these shortcomings, an effective matrix representation for hypergraphs is needed, such that the ambiguities of relational order can be overcome. To this end, trivial graph approximations should be avoided for hypergraph representation. In the mathematics literature, the definitions of the Ihara zeta function has recently been extended from graphs to hypergraphs [91]. In the determinant form of the Ihara zeta function, a graph representation is also used for describing the hypergraph. However, this graph representation uses color edges to capture the hyperedge connectivity and does not result in information loss regarding relational order. We will make a polynomial analysis of the hypergraph Ihara zeta function and develop a family of features that readily characterize hypergraphs into a feature space suitable for hypergraph clustering.

2.4 Structural Matching

Many problems in computer vision and machine learning are those of establishing consistent correspondences between two sets of features. This is possible because individual feature sets usually have internal structure, and between-set correspondences can be recovered by measuring the (dis)similarities between the structural contexts of the features. We refer to the process of establishing correspondences subject to structural similarity as structural matching, which is necessary in many pattern recognition tasks. For example, to establish visual correspondences between two objects from the same class, we need not only to measure the visual similarities but also maintain the consistence in spatial arrangement between corresponding feature points. In addition to the straightforward applications, challenges of structural matching also arise in knowledge discovery from

large graph datasets, where suitable methods are required to robustly organize, query and navigate large sets of graphs extracted from real world data, such as visual images and chemical molecules. To this end, we need to establish an explicit class archetype and maintain correspondences between individual objects belonging to it. Additionally, the procedure of correspondence maintenance is also of key importance in learning generative models for graphs, where vertex correspondences are assumed to be prior knowledge for constructing supergraph prototypes [93].

The early research literature on structural matching focuses on establishing consistent correspondences between features using pairwise relationships. This problem is often formulated in terms of graph matching, which can be solved by optimizing an objective function associated with the compatibility matrix (i.e. the matrix with entries representing the agreements of two graph edges). Umeyama [95] has presented a spectral matching method for graphs of the same size. In his study, a pair of graphs both with vertex cardinality N are embedded into feature space where an N-dimensional coordinate is obtained for each vertex. Comparisons are made between every pair of vertex coordinates and the Hungarian algorithm is used to extract the most dominant correspondences. Based on a similar notion, Shapiro *et al.* [82] have proposed a spectral technique which accommodates graphs with different sizes by truncating the least significant modes in the coordinate vectors for vertices. Gold *et al.* [35] have provided a softassign strategy, in which a two-way constraint is enforced in the objective function and Sinkhorn's normalization method is used to obtain the optimal solution. Wilson *et al.* [101] have derived a probability distribution for relational errors that occur when there is significant graph corruption and presented a graph matching method based on discrete relaxation. Luo *et al.* [58] have presented a graph matching approach by using the EM algorithm and singular value decomposition. An overview of the research literature before the year 2004 can be found in the survey article [24]. Recently, Leordeanu *et al.* [50] have proposed an efficient matching approach based on graph spectral methods, in which the elements

of the principal eigenvector for the compatibility matrix are used as the matching scores. Further developments of this graph spectral approach include balancing the compatibility matrix [25] and developing an efficient algorithm for solving quadratic assignment problems in graph matching [52]. Albarelli *et al.* [3] have formulated the matching problem in terms of a non-competitive game. Furthermore, various learning methods have been proposed to optimize the parameters for graph matching [52][16].

All the graph matching methods are confined to pairwise representations, i.e. the relational order between features is two. Our work in this thesis, on the other hand, is concerned with high order structural matching, i.e. relational orders not restricted to two. We will investigate the problem of high order structural matching in terms of hypergraph formulations. Hypergraphs have the capability of capturing multiple relationships between features. After presenting one hypergraph characterization method in Chapter 4, we will continue to explore hypergraphs in a structural matching scenario in Chapter 5, since characterization and matching are the two main strategies for learning classes of structural patterns.

Recently a number of researchers have attempted to extend the matching process to incorporate higher order relations. Their research aims are similar to those in this thesis. Zass *et al.* [110] are among the first to investigate this problem by introducing a probabilistic hypergraph matching framework, in which higher order relationships are marginalized to unary order. Chertok *et al.* [21] have improved this work by marginalizing the higher order relationships to be pairwise and then adopted pairwise graph matching methods. However, these two methods only approximate the hypergraph representation by using a clique graph. It has already been pointed out in [1] that this graph approximation is just a low pass representation of the original hypergraph and causes information loss and inaccuracy. On the other hand, Duchenne *et al.* [28] have developed the spectral technique for graph matching [50] into a higher order matching framework using *tensor power iteration*. Although they adopt an L_1 norm constraint in computation, the origi-

nal objective function is subject to an L_2 norm. Furthermore, their method is developed in a heuristic manner, and the effectiveness can not be clearly explained. Our proposed method will address these shortcomings accordingly. We do not approximate hypergraphs using graph representation. Moreover, our proposed method will satisfy the basic axiom of probability and can be explained in terms of high order clustering.

2.5 Summary

We have reviewed the research literature on structural characterization and matching, both for pairwise and higher order representations. We have analyzed the deficiencies of the existing methods and pointed out our possible solutions for overcoming these shortcomings. This chapter can be summarized as follows.

There is a substantial body of research on graph based learning methods and most of it lies in the graph partition or graph embedding scenario. Graphs as natural representations for structured data are also widely used in the structural pattern recognition. In the literature most structural characteristics are extracted from the original form or the original matrix representation of a graph, and graph characterization in a transformed domain has rarely been investigated. The Ihara zeta function provides potential approaches to characterizing transformed graphs using polynomial or spectral methods. Furthermore, the original form of the Ihara zeta function is defined on cycle frequencies and thus is closely related to graph topologies. These properties of the Ihara zeta function motivate us to explore it as a powerful structural characterization method.

The research literature on hypergraph based learning algorithms is generally confined to tensor factorization, which is a higher order extension of its pairwise counterpart. When hypergraphs are used for representing higher order structured data in structural pattern recognition, they are often approximated by a graph representation. Trivial graph approximations may give rise to certain information loss and result in ambiguities in distinguish-

ing different relational orders. To address these shortcomings in the existing hypergraph-based methods, we will continue to explore the Ihara zeta function as an effective tool for high order characterization. We will also show that the characterization method developed in this thesis is flexible for both pairwise and higher order structures and can thus overcome the ambiguities caused by the approximation methods in the literature.

Research on structural matching has extended over forty years and various strategies have been proposed to solve the pairwise correspondence problems. In contrast, the research on high order matching is still at a relatively early stage. Existing high order matching algorithms are either approximated by graph matching or developed in a heuristic manner, and thus are far from maturity. To address these shortcomings, we will present a novel hypergraph matching framework. Our method can be explained in terms of high order clustering and we will show its advantages over the state of the art methods.

Above all, the work in this thesis addresses the shortcomings in the research literature and aims to present effective structural characterization and matching methods. We will compare our proposed methods with the state of the art methods and discuss in detail our contributions to the research literature in the subsequent chapters.

Chapter 3

Graph Characterization via Ihara Coefficients

We present a route to graph characterization based on the Ihara zeta function in this chapter. We first review the fundamental knowledge related to the Ihara zeta function and describe how to extract graph characteristics from it. We then extend the representation to edge-weighted graphs and provide a spectral interpretation of the proposed method. Both quantitative and qualitative experiments are conducted to verify the effectiveness of the proposed method in graph characterization.

3.1 The Ihara Zeta Function

In this section we commence by reviewing how to construct the Ihara zeta function for a graph. We then explore the possibility of using a set of polynomial coefficients from the Ihara zeta function as a representation of graph-structure. Our novel contribution here is to use the existing ideas from graph theory to develop a new representation for graphs.

3.1.1 Definition

In a graph a cycle (i.e. closed path) is referred to as a prime cycle as long as it satisfies the following two conditions: i) the cycle and its cyclic permutations have no backtracking; ii) the cycle is not a multiple of another cycle. The equivalence class $[p]$ associated with the prime cycle p is the set of cyclic permutations of p. Based on these notions, the Ihara zeta function associated with a finite connected graph is defined as [44][45]:

$$Z_G(u) = \prod_{[p]} \left(1 - u^{L(p)}\right)^{-1}. \tag{3.1}$$

The product is over the equivalence classes $[p]$ of prime cycles p, and $L(p)$ denotes the length of p. Since the number of prime cycles in a graph is usually infinite, the expression given in (3.1) is not tractable for practical purposes. As a result, the Ihara zeta function is generally an infinite product. However, one of its elegant features is that it can be collapsed down into a rational function, which renders it of practical utility.

3.1.2 Rational Expression

For a graph $G(V, E)$ with vertex set V of cardinality $|V| = N$ and edge set E of cardinality $|E| = M$, the rational expression for the Ihara zeta function is [7]:

$$Z_G(u) = \left(1 - u^2\right)^{\chi(G)} \det\left(\boldsymbol{I}_N - u\boldsymbol{A} + u^2\boldsymbol{Q}\right)^{-1}, \tag{3.2}$$

where $\chi(G)$ is the Euler number of the graph $G(V, E)$, which is defined as $\chi(G) = N - M$, and $\boldsymbol{A}$ is the adjacency matrix of the graph. The degree matrix $\boldsymbol{D}$ is constructed by placing the column sums of the adjacency matrix as diagonal elements, while setting the off-diagonal elements to zero. Finally, with $\boldsymbol{I}_k$ denoting the $k \times k$ identity matrix, $\boldsymbol{Q}$ is the matrix difference of the degree matrix $\boldsymbol{D}$ and the identity matrix $\boldsymbol{I}_N$, i.e. $\boldsymbol{Q} = \boldsymbol{D} - \boldsymbol{I}_N$.

3.1.3 Permutation Invariant

The topology of a graph is invariant under permutations of the node labels. However, neither the adjacency matrix nor the Laplacian matrix for isomorphic graphs are permutation invariant. Suppose $\boldsymbol{A}$ and $\boldsymbol{A}_p$ are the adjacency matrices associated with isomorphic graphs $G(V,E)$ and $G_P(V,E)$ respectively. Let $\boldsymbol{P}$ be the permutation matrix representing the changes in node order between $G(V,E)$ and $G_P(V,E)$. $\boldsymbol{P}$ is an orthogonal matrix. The permutation relationship between the two adjacency matrices is $\boldsymbol{A}_P = \boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^T$ and so is $\boldsymbol{Q}_P = \boldsymbol{P}\boldsymbol{Q}\boldsymbol{P}^T$. The Ihara zeta function of $G_P(V,E)$ can be denoted as:

$$\begin{aligned} Z_{G_P}(u) &= \left(1-u^2\right)^{\chi(G)} \det\left(\boldsymbol{I} - u\boldsymbol{A}_P + u^2\boldsymbol{Q}_P\right)^{-1} \\ &= \left(1-u^2\right)^{\chi(G)} \det\left(\boldsymbol{I} - u\boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^T + u^2\boldsymbol{P}\boldsymbol{Q}\boldsymbol{P}^T\right)^{-1} \\ &= \left(1-u^2\right)^{\chi(G)} \det\left(\boldsymbol{P}\boldsymbol{I}\boldsymbol{P}^T - u\boldsymbol{P}\boldsymbol{A}\boldsymbol{P}^T + u^2\boldsymbol{P}\boldsymbol{Q}\boldsymbol{P}^T\right)^{-1} \\ &= \left(1-u^2\right)^{\chi(G)} \det\left(\boldsymbol{I} - u\boldsymbol{A} + u^2\boldsymbol{Q}\right)^{-1} \\ &= Z_G(u). \end{aligned} \tag{3.3}$$

The Ihara zeta functions of the graphs G and G_P appearing in (3.3) have the same form. This proves that it is invariant to node order permutations. As a result, the equivalent representations of the Ihara zeta function, such as its derivatives or the coefficients of the polynomial of its reciprocal, are also invariant to node order permutation.

3.1.4 Determinant Expression

For md2 graphs, i.e. those graphs in which every vertex has degree at least 2, it is straightforward to show from equation (3.1) that the Ihara zeta function can be rewritten in the form of the reciprocal of a polynomial. Although the Ihara zeta function can be evaluated efficiently using (3.2), the task of enumerating the coefficients of the polynomial appearing in the denominator of the Ihara zeta function (3.2) is difficult, except by resorting to software for symbolic calculation. To efficiently compute these coefficients we adopt a

different strategy. We first transform the graph into an oriented line graph. The Ihara zeta function is then the reciprocal of the quasi characteristic polynomial for the adjacency matrix of the oriented line graph.

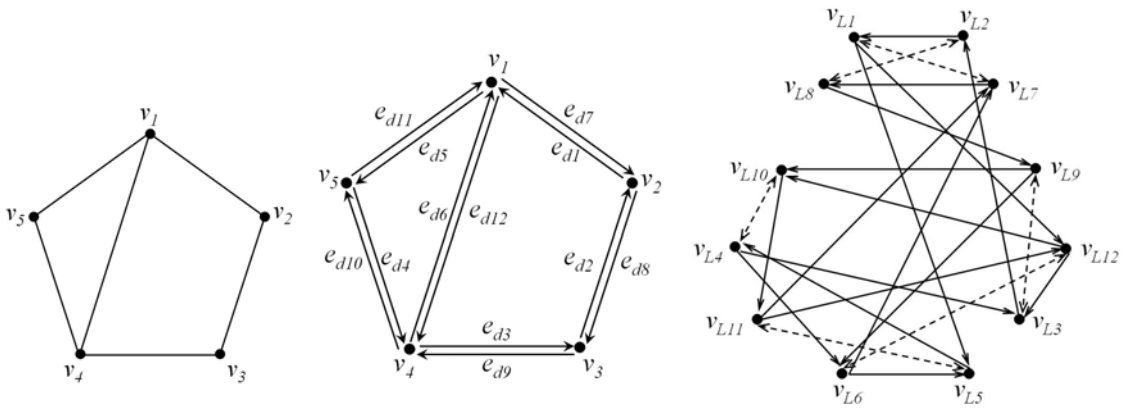
Oriented Line Graph

To commence, we construct the oriented line graph of the original graph from the associated symmetric digraph. The symmetric digraph $SDG(V, E_d)$ of a graph $G(V, E)$ is composed of a finite nonempty vertex set V identical to that of $G(V, E)$ and a finite multiset E_d of oriented edges called arcs, which consist of ordered pairs of vertices. For arc $e_d(u, w) \in E_d$ where u and w are elements in V, the origin of $e_d(u, w)$ is defined to be $o(e_d) = u$ and the terminus is $t(e_d) = w$. Its inverse arc, which is formed by switching the origin and terminus of $e_d(u, w)$, is denoted by $e_d(w, u)$. For the graph $G(V, E)$, we can obtain the associated symmetric digraph $SDG(V, E_d)$ by replacing each edge of $G(V, E)$ by an arc pair in which the two arcs are inverse to each other. An example illustrating how to derive the symmetric digraph from an original graph is shown in Figure 3.1. Figure 3.1(a) shows an example graph with five vertices and six edges and Figure 3.1(b) shows its corresponding symmetric digraph.

The oriented line graph associated with the original graph can be defined using the symmetric digraph. It is a dual graph of the symmetric digraph since its arc set and vertex set are constructed from the vertex set and the arc set of its corresponding symmetric digraph, respectively. The construction of the vertex set V_L and the arc set E_{dL} of the oriented line graph can be expressed as follows [49]:

$$\begin{cases} V_L \ = E_d(SDG); \\ E_{dL} = \{(e_d(u, v), e_d(v, w)) \in E_d(SDG) \times E_d(SDG);\ u \neq w\}. \end{cases} \tag{3.4}$$

From (3.4) we can see that the vertices of the oriented line graph are obtained from the arcs of the symmetric digraph, and as a result we can denote an arc e_{di} in the symmetric digraph and its corresponding vertex v_{Li} in the oriented line graph using the same index



(a) Original graph. (b) Symmetric digraph. (c) Oriented line graph.

FIGURE 3.1: GRAPH EXAMPLE.

i. According to (3.4), an arc would be directed from vertex v_{Li} to v_{Lj} in the oriented line graph subject to the necessary condition that the terminus of the arc e_{di} connects the origin of the arc e_{dj}. The only exception in this case is that the reverse arc pairs do not establish arcs in the oriented line graph. For the example graph in Figure 3.1(a), the corresponding oriented line graph is shown in Figure 3.1(c) (disregarding for the time being the arcs consisting of a dashed line and two arrows on both ends). We can see that there is an arc directed from vertex v_{L11} to v_{L7} in Figure 3.1(c), due to the connection between the terminus of the arc e_{d11} and the origin of the arc e_{d7} in Figure 3.1(b). However, there is no arc between the vertex v_{L3} and v_{L9} in Figure 3.1(c), because the arcs e_{d3} and e_{d9} are a reverse arc pair in Figure 3.1(b).

The adjacency matrix $\boldsymbol{T}$ of the oriented line graph, which is a $2M \times 2M$ square matrix, is referred to as the Perron-Frobenius operator of the original graph. For the (i, j)th entry of $\boldsymbol{T}$, $\boldsymbol{T}(i, j)$ is 1 if there is one arc directed from the vertex i to the vertex j in the oriented line graph, and 0 otherwise. Unlike the adjacency matrix of an undirected graph, the Perron-Frobenius operator $\boldsymbol{T}$ is not a symmetric matrix due to the constraint that arises in (3.4).

Furthermore, we also notice that the arc set of the oriented line graph is adopted by

discrete-time quantum walks as the state space. To make this study one step further, we investigate the relationship between the Ihara zeta function and the discrete-time quantum walks, and find that the Perron-Frobenius operator can be formulated in terms of the positive support of the transpose of the transition matrix of the discrete-time quantum walks. Detailed analysis of this relationship can be found in Appendix A.

Quasi Characteristic Polynomial

With the oriented line graph to hand, the Ihara zeta function can be written in the form of a determinant [7][49] using the Perron-Frobenius operator:

$$Z_G^{-1}(u) = \det(\boldsymbol{I} - u\boldsymbol{T}). \tag{3.5}$$

Here we refer to $\det(\boldsymbol{I} - \lambda\boldsymbol{M})$ as the quasi characteristic polynomial of the matrix $\boldsymbol{M}$. It is different from the characteristic polynomial $\det(\lambda\boldsymbol{I} - \boldsymbol{M})$ by multiplying the argument with $\boldsymbol{M}$ rather than the identity matrix $\boldsymbol{I}$. From (3.5) it is clear that the reciprocal of the Ihara zeta function for a graph is the quasi characteristic polynomial of $\boldsymbol{T}$, and can be expressed as:

$$Z_G^{-1}(u) = c_0 + c_1 u + \cdots + c_{2M-1}u^{2M-1} + c_{2M}u^{2M}. \tag{3.6}$$

We refer to the polynomial coefficients $\{c_1, c_2, \ldots, c_{2M}\}$ in (3.6) as Ihara coefficients. They are the coefficients of the quasi characteristic polynomial and are the antitone sequence of the characteristic polynomial coefficients of $\boldsymbol{T}$.

3.1.5 Ihara Coefficients for Characterizing Graphs

To establish pattern vectors from the Ihara zeta function for the purpose of machine learning, it is natural to consider taking function-samples as the elements of the pattern vectors. However, function-samples have no known exact significance related to graph structure. Furthermore, there is the possibility of sampling at the locations of poles and these give

rise to infinities. Hence, the pattern vectors consisting of function samples are potentially unstable representations of graphs.

On the other hand, the Ihara coefficients do not give rise to infinities. These coefficients are essentially descriptors of graph structure. Provided $G(V, E)$ is a simple graph, the coefficients have the following properties [79]: a) the coefficients c_3, c_4, and c_5 are respectively the negative of twice the number of triangles, squares, and pentagons in G, b) c_6 is the negative of twice the number of hexagons in G plus four times the number of pairs of edge disjoint triangles plus twice the number of pairs of triangles with a common edge, c) c_7 is the negative of twice the number of heptagons in G plus four times the number of edge disjoint pairs of one triangle and one square plus twice the number of pairs of one triangle and one square that share a common edge, and d) the highest order coefficient is associated with the degree $d(v_i)$, which is the number of edges incident to vertex v_i:

$$c_{2M} = (-1)^{\chi(G)} \prod_{v_i \in V} \left(d(v_i) - 1\right). \tag{3.7}$$

For unweighted graphs, the set of Ihara coefficients can play the role of pattern vectors for describing graphs, not only because they are numerically stable, but also because they characterize the graph structure in terms of cycle frequencies.

3.2 Ihara Coefficients for Edge-weighted Graphs

Although (3.5) characterizes unweighted graphs in a compact way using the quasi characteristic polynomial of the Perron-Frobenius operator, when it comes to edge-weighted graphs, i.e. the edges not only record the vertex connections but also have a scalar weight attached to each of them, then the method introduced in Section 3.1.4 is no longer applicable. This is because the Perron-Frobenius operator for an unweighted graph is the adjacency matrix of the associated oriented line graph, which only bears relational information and defaults the edge weights to binary values. This hinders the generalization

of the determinant expression of the Ihara zeta function as a characterization of edge-weighted graphs.

To overcome this problem and furnish characterization methods for edge-weighted graphs, we generate the required Perron-Frobenius operator from a simplified version of the Bartholdi zeta function. The Bartholdi zeta function is a more sophisticated zeta function defined over two independent variables. We then introduce a scheme to characterize the edge-weighted graphs using the quasi characteristic polynomial of the modified Perron-Frobenius operator. The novel contribution here is to extend the definition of Ihara coefficients from unweighted to edge-weighted graphs.

3.2.1 Bartholdi Zeta Function

The Bartholdi zeta function of a graph aims to generalize the zeta function using the concept of a 'prime circle' [6]. For a graph $G(V, E)$, the rational expression of the Bartholdi zeta function is:

$$\begin{aligned} Z_{GB}(u,t) =& \left(1-(1-t)^2u^2\right)^{\chi(G)} \times \\ & \det\left(\boldsymbol{I}_N - u\boldsymbol{A} + (1-t)u^2(\boldsymbol{D}-(1-t)\boldsymbol{I}_N)\right)^{-1}, \end{aligned} \tag{3.8}$$

where $\chi(G)$ is the Euler number (defined with (3.2) previously for the Ihara zeta function). Compared with the Ihara zeta function in its rational form (3.2), the rational expression for the Bartholdi zeta function involves an additional variable t, which is closely related to the cyclic bump count of a cycle in a graph. Here a cycle with one-step backtracking is referred to as a single cyclic bump. One noteworthy property of the Bartholdi zeta function is that when $t = 0$, it reduces to the Ihara zeta function. For more details of the Bartholdi zeta function and its relationship with the Ihara zeta function, we refer readers to [6], [61] and [77].

3.2.2 Ihara Polynomial for Edge-weighted Graphs

Based on Bartholdi's work, Mizuno and Sato have developed the following determinant expression for the Bartholdi zeta function [61]:

$$Z_{GB}(u,t) = \det\left(\boldsymbol{I}_{2M} - (\boldsymbol{B} - (1-t)\boldsymbol{J})u)\right)^{-1}. \tag{3.9}$$

In this form the zeta function is equivalent to (3.8) and is suitable for dealing with both unweighted graphs and edge-weighted graphs. There are two $2M \times 2M$ operators $\boldsymbol{B}$ and $\boldsymbol{J}$ in (3.9), which are both closely related to the symmetric digraph of the original graph. For a graph $G_w(V,E)$ with weighted edges, the associated symmetric digraph $SDG_w(V,E)$ can be constructed by replacing each edge of $G_w(V,E)$ with an arc pair in which the two arcs are the reverses of each other. The edge weights are then attached to each of its generating arcs. This process is quite similar to that adopted in the case of unweighted graphs introduced in Section 3.1.4, except that there is the additional step of weight attachment. To illustrate this point, suppose that the example in Figure 3.1(a) is edge-weighted. Its symmetric digraph will retain the structure shown in the Figure 3.1(b). The only difference is that there is an attribute associated to each reverse arc pair determined by the edge weight in the original graph.

Based on the symmetric digraph, the elements of the operators $\boldsymbol{B}$ and $\boldsymbol{J}$ can be computed as follows:

$$\boldsymbol{B}_{ij} = \begin{cases} w_{dj} & \text{if } t(e_{di}) = o(e_{dj}) \\ 0 & \text{otherwise,} \end{cases} \quad \boldsymbol{J}_{ij} = \begin{cases} 1 & \text{if } e_{di} = \bar{e}_{dj} \\ 0 & \text{otherwise.} \end{cases} \tag{3.10}$$

Here, a) $t(e_{di})$ and $o(e_{dj})$ denote the origin and terminus of the arc e_{di} in the symmetric digraph, respectively, b) $\bar{e}_{dj}$ denotes the inverse arc of e_{dj} and c) w_{dj} is the weight of the arc e_{dj} in the symmetric graph, which is equal to the weight of the edge of the original graph from which the arc is derived.

For edge-weighted graphs, when $t = 0$ in (3.9), the determinant form of the Bartholdi

zeta function reduces to the corresponding Ihara form denoted as follows:

$$Z_G(u) = \det(\boldsymbol{I}_{2M} - u(\boldsymbol{B} - \boldsymbol{J}))^{-1} = \det(\boldsymbol{I}_{2M} - u\boldsymbol{T}_w)^{-1}. \tag{3.11}$$

where we define $\boldsymbol{T}_w = \boldsymbol{B} - \boldsymbol{J}$ to be the Perron-Frobenius operator for edge-weighted graphs. Moreover, $\boldsymbol{T}_w$ can also be regarded as the adjacency matrix of the oriented line graph of the original edge-weighted graph. The form in (3.11) was mentioned in [77] as an intermediate step to develop a new Bartholdi zeta function. However, it has not been studied from the perspective of edge-weighted graphs, which are the emphasis of this section.

The establishment of the matrices $\boldsymbol{B}$ and $\boldsymbol{J}$ summarizes the process of constructing the oriented line graph from a edge-weighted graph. Here, the oriented line graph is similar in structure with that for the corresponding unweighted case, except with the addition of arcs between vertices deriving from the reverse arc pairs in the symmetric digraph. Unlike the unweighted case, the reverse arc pairs in the symmetric digraph are used to establish arcs in the oriented line graph from an edge-weighted graph. That is, if the arcs e_{di} and e_{dj} are a reverse pair in the weighted symmetric digraph, there would be a reverse arc pair between the vertices v_{Li} and v_{Lj} in the oriented line graph. The weight measures of the reverse arc pairs are different from the remaining arcs in the oriented line graph. Suppose that the vertices in the oriented line graph are attributed by the same weights as their corresponding arc in the symmetric digraph. Each edge in a reverse arc pair is attributed by the weight of its terminus vertex minus one, according to (3.10) and (3.11). The remaining arcs that have no reverse counterparts are attributed by their terminus vertex weights only, according to the definition of $\boldsymbol{B}$ in (3.10).

The structure of the oriented line graph of the example graph in Figure 3.1(a) (suppose it is edge weighted) is shown in Figure 3.1(c). A dashed line with arrows at both ends is an arc pair, which represents two arcs that are reverse to each other and originate from the reverse arcs in Figure 3.1(b). Solid lines with an arrow on one end and which are similar part to the unweighted case in structure, are arcs derived subject to the constraint

in (3.4). The arcs denoted by the dashed line and the solid line are weighted differently. For example, the edge directed from the vertex v_{L11} to v_{L7} is attributed by the edge-weight of e_{d7} in Figure 3.1(b), that is, the weight of the edge connecting v_1 and v_2 in Figure 3.1(a). On the other hand, each edge of the arc pair between the vertices v_{L4} and v_{L10} is attributed by the edge-weight of e_{d4} or e_{d10} in Figure 3.1(b) minus one, that is, the weight of the edge connecting v_4 and v_5 in Figure 3.1(a) minus one.

There are several points that need to be stressed here. First, unlike in (3.4), there is no requirement on the exclusion of reversed arcs in the computation of the operator $\boldsymbol{B}$ in (3.10). Second, the operator $\boldsymbol{J}$ records all the reversed arc relations. Third, when it comes to unweighted graphs, $\boldsymbol{T}_w$ reduces to $\boldsymbol{T}$ in (3.5). This is because in this case $\boldsymbol{B}$ reduces to a binary matrix representing orientations and connections only. More importantly, the operator $\boldsymbol{T}_w$, i.e. the difference of $\boldsymbol{B}$ and $\boldsymbol{J}$, naturally satisfies all the constraints in (3.4) when it is reduced to an unweighted version. Above all, our proposed scheme establishes a generalized version of the Perron-Frobenius operator, which is available to both unweighted graphs and edge-weighted graphs.

3.2.3 Ihara Coefficients for Edge-weighted Graphs

According to our observations in Section 3.2.2, the Ihara coefficients can be constructed for both unweighted graphs and edge-weighted graphs in a unified manner. Here, we denote the Perron-Frobenius operator of a graph, irrespective of whether it is unweighted or edge-weighted, as $\boldsymbol{T}_w$. According to (3.11), the Ihara coefficients can be derived from the polynomial:

$$\begin{aligned} Z_G^{-1}(u) &= \det(\boldsymbol{I} - u\boldsymbol{T}_w) \\ &= c_0 + c_1 u + \cdots + c_{2M-1}u^{2M-1} + c_{2M}u^{2M}. \end{aligned} \tag{3.12}$$

From (3.12), the Ihara coefficients $\{c_0, c_1, \ldots, c_{2M}\}$ are the coefficients of the quasi characteristic polynomial of the matrix $\boldsymbol{T}_w$. Pattern vectors $\vec{v} = [c_0\ c_1\ \ldots\ c_{2M}]^T$ for character-

izing graphs can then be established with Ihara coefficients as components. As discussed in Section 3.1.5, the Ihara coefficients not only avoid the hazards of infinities that are encountered if function samples are used, but also convey information concerning graph structure and topology.

3.3 Graph Spectral Interpretation

For unweighted graphs, the Ihara coefficients are essentially descriptors of graph structure. For edge-weighted graphs, however, the Ihara coefficient have no direct links with the cycle frequencies or vertex degrees in the graph. Here we study the Ihara coefficients for md2 graphs from a spectral standpoint. We perform a comprehensive analysis on the effectiveness of the Ihara coefficient for clustering md2 graphs.

As stated in Sections 3.1.4 and 3.2.2, there is always one oriented line graph associated with an md2 graph. In practice the cardinality of the vertex set of the oriented line graph is much greater than, or at least equal to, that of the original graph. The construction of the oriented line graph is thus a process of transforming the original graph into a new version with adjacency matrix $\boldsymbol{T}_w$ in a higher dimensional space than that of the original graph. Furthermore, the Ihara coefficients are determined by the spectrum of the Perron-Frobenius operator. Each coefficient can be derived from the polynomial of the eigenvalue set $\{\tilde{\lambda}_1, \tilde{\lambda}_2 \ ... \ \tilde{\lambda}_n\}$ of $\boldsymbol{T}_w$ as follows:

$$c_{2M-r} = (-1)^r \sum_{k_1<k_2< \ ... \ <k_r} \tilde{\lambda}_{k_1} \tilde{\lambda}_{k_2} \ ... \ \tilde{\lambda}_{k_r} \tag{3.13}$$

The eigenvalue set of $\boldsymbol{T}_w$ is just the pole set of the Ihara zeta function. For an unweighted graph, it is linked to the lengths of prime cycles. Equation (3.13) provides an efficient way to compute the Ihara coefficients by enumerating the eigenvalues of a $2M \times 2M$ matrix. This is close in spirit to the first method for computing the polynomial coefficients suggested by Brooks [12]. Moreover, it is more efficient than the third method suggested

by Brooks, which is based on computing the determinant of $2M \times 2M$ matrix $\binom{2M}{2M-r}$ times to obtain the coefficient c_{2M-r}.

From the above spectral analysis, we can see that the computational complexity of obtaining the Ihara coefficients is governed by the eigen-decomposition of the Perron-Frobenius operator and this requires $O(M^3)$ operations.

The Ihara coefficients are signed elementary symmetric polynomials (ESP) on the eigenvalues of $\boldsymbol{T}_w$. It is interesting to note that Wilson, Luo and Hancock [102] have applied elementary symmetric polynomials to the elements of the spectral matrix $\boldsymbol{Y} = \sqrt{\boldsymbol{\Lambda}}\boldsymbol{\Phi}^T$ of the Laplacian matrix $\boldsymbol{L} = \boldsymbol{Y}\boldsymbol{Y}^T = \boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^T$. The same authors have also explored using the leading eigenvalues of the graph matrix representations, i.e. truncated spectra [58]. By using ESP they overcome problems of restricted performance encountered using the truncated spectra. However, the work is based on the Laplacian matrix $\boldsymbol{L}$ rather than the Perron-Frobenius operator $\boldsymbol{T}_w$. The Ihara coefficients, on the other hand, are the coefficients of the quasi characteristic polynomial of $\boldsymbol{T}_w$ and are determined by the full spectrum of $\boldsymbol{T}_w$ according to (3.13). Furthermore, the Perron-Frobenius operator extracts graph characteristics in a higher dimensional feature space instead of the matrix representations of the original graph. Therefore, the Ihara coefficients not only capture richer graph characteristics than graph spectral representations, but also naturally avoid the spectral truncation. In our experiments, we compare the performance of the Ihara coefficients with the truncated Laplacian spectrum and the ESP of the Laplacian spectrum.

3.4 Experimental Evaluation

We test our proposed method on both unweighted and edge-weighted graphs. For graphs of each type, we report results on both synthetic and real world data. The experiments on synthetic data are aimed at evaluating the ability of the Ihara coefficients to distinguish between graphs under controlled structural errors. The experiments on real world data

aim to assess whether the Ihara coefficients can be embedded in a pattern space that reveals their cluster structure. Both qualitative and quantitative evaluations are made in our experiments.

3.4.1 Unweighted Graphs

Synthetic Graphs

We commence by investigating the relationship between graph edit distance and the Euclidean distance between pattern vectors composed of Ihara coefficients. The edit distance between two graphs G_1 and G_2 is the minimum edit cost taken over all sequences of edit operations that transform G_1 into G_2. In our experiments we establish a new graph by deleting a certain number of edges from a seed graph. We assign to each deletion an edit cost equal to the deleted edge weight. The edit distance between the edited graph and the seed graph is equal to the total deleted edge weights, i.e. $d_{Edit} = \sum_{e\ Deleted} w(e)$. For unweighted graphs, the edge weight is unity for each deleted edge and the edit distance is equal to the number of edges deleted.

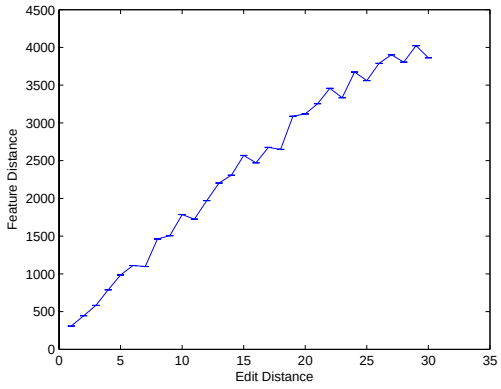
We commence with a set of randomly generated md2 graphs. The seed graph for the set has 100 vertices and 300 edges. The remainder of the graphs in the set are obtained by deleting the edges of the seed graph (indexed from 1 to 30). At each level of editing, 100 trials are performed and the edges deleted are chosen randomly, subject to preserving the md2 constraint. In our experiments, we construct two Ihara coefficient pattern vectors $\vec{v}_I = [c_3,\ c_4,\ c_5,\ c_6,\ c_7,\ \ln(|c_{2M}|)]^T$ and $\vec{v}_{IS} = [c_3,\ c_4,\ \ln(|c_{2M}|)]^T$. Since the components of $\vec{v}_{IS}$ are a subset of those of $\vec{v}_I$, we refer to $\vec{v}_I$ as the pattern vector composed of unselected Ihara coefficients and $\vec{v}_{IS}$ as that composed of selected Ihara coefficients. The final component of each pattern vector is scaled in a logarithmic manner to avoid problems of dynamic range. For comparison, we also establish pattern vectors with the same dimension as $\vec{v}_I$ using the truncated spectrum of the Laplacian matrix.

The leading eigenvalues of the Laplacian matrix experimentally play a dominant role in characterizing graphs. Thus we take the second smallest through to the seventh smallest eigenvalues of graph Laplacian $\boldsymbol{L}$ as components, i.e. $\vec{v}_L = [\lambda_2,\ \lambda_3,\ \lambda_4,\ \lambda_5,\ \lambda_6,\ \lambda_7]^T$, where $\boldsymbol{L} = \boldsymbol{\Phi}\boldsymbol{\Lambda}\boldsymbol{\Phi}^T$, $\boldsymbol{\Lambda} = diag(\lambda_1,\ \lambda_2,\ \cdots\ \lambda_N)$ and $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N$. We refer to $\vec{v}_L$ as the Laplacian spectral pattern vector, which proves to be one of the most effective spectral representations for graphs [103]. The feature distance between pattern vectors $\vec{v}_i$ and $\vec{v}_j$ is defined as the Euclidean distance $d_{i,j} = \sqrt{(\vec{v}_i - \vec{v}_j)^T(\vec{v}_i - \vec{v}_j)}$. We investigate the representational power and stability of the pattern vectors. The experimental results are shown in Figure 3.2. Figure 3.2(a) shows the feature distance between pattern vectors (composed of unselected Ihara coefficients) of the seed graph and the edited graph as a function of the edit distance, i.e. the number of edges deleted. The relative standard deviation (RSD) is also shown as an error bar on each measurement. The Ihara coefficient distance generally follows the edit distance. Figure 3.2(b) shows the feature distance between Laplacian spectral pattern vectors, as well as the corresponding RSD, as a function of the edit distance (but scaled differently to Figure 3.2(a)). From Figures 3.2(a) and 3.2(b) it is clear that the dynamic range of the coefficient feature distance is much larger than that of the Laplacian spectral distance for a corresponding edit distance. Thus the Ihara coefficients are more sensitive to graph edits than the truncated Laplacian spectra. Figure 3.2(c) shows the feature distance computed using the selected Ihara coefficients as a function of the edit distance. Compared with Figure 3.2(a), the selected Ihara coefficients are more linear with the edit distance than the unselected Ihara coefficients. To evaluate how reliably the Ihara coefficient distance and the Laplacian spectral distance predict the edit distance, we plot their RSD as a function of edit distance in Figure 3.2(d). From Figure 3.2(d), it is clear that although the unselected Ihara coefficients are less stable than the graph spectra, the selected Ihara coefficients offer the best stability. This illustrates that the Ihara coefficients have the potential to provide a more stable representation than the truncated Laplacian spectra when the proper coefficient selection is performed.

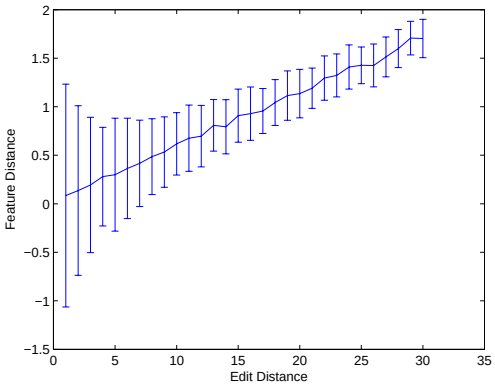
Further discussion on the coefficient selection will be detailed in the next subsection.

View-based Object Recognition

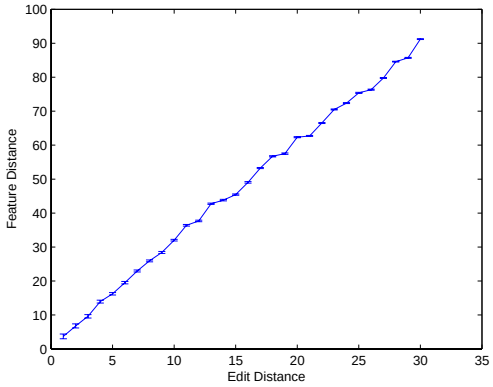
We apply the pattern vectors composed of Ihara coefficients to two graph datasets. The first set of graphs are extracted from three sequences of images of model houses [58]. The second set of graphs are extracted from images of three toys. The third set of graphs are extracted from images of the objects in the COIL database [62]. Sample images of the model houses, toys and objects in the COIL database are shown in Figures 3.3(a) (referred to as the CMU, MOVI and Chalet sequences from left to right), 3.3(b) and 3.3(c) respectively. There are ten pictures taken for each model house from different view angles, thirty pictures taken for each toy from different view angles, and seventy two pictures taken for each object in the COIL database from different view angles. We first extract corner points from each image using the Harris detector [39]. Then we establish Delaunay graphs based on these corner points as vertices. Each vertex is used as the seed of a Voronoi region, which expands radically with a constant speed. The collision fronts of the regions delineate the image plane into polygons, and the Delaunay graph is the region adjacency graph for the Voronoi polygons. Figure 3.3 shows the Delaunay graphs superimposed on the sample images of the objects. Once we have extracted graphs from the objects, we can establish pattern vectors using either the Ihara coefficients or the truncated Laplacian spectra. One way to evaluate the effectiveness of the pattern vectors for clustering graphs is to apply K-means to them to obtain clusters and then compute the Rand indices to assess the clustering performance. The Rand index is defined as $R_I = X/(X+Y)$, where X is the number of agreements and Y is the number of disagreements in cluster assignment. If two objects are in the same cluster in both the ground truth clustering and the clustering from our experiment, this counts as an agreement. If two objects are in the same cluster in the ground truth clustering but are in different clusters in our experiment, this counts as a disagreement. The Rand index takes a value in the



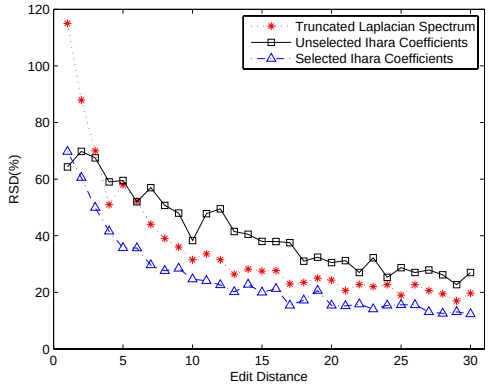
(a) Unselected Ihara coefficients.



(b) Truncated Laplacian spectra.

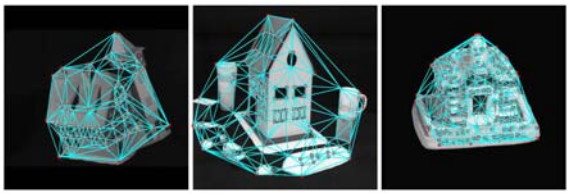


(c) Selected Ihara coefficients.

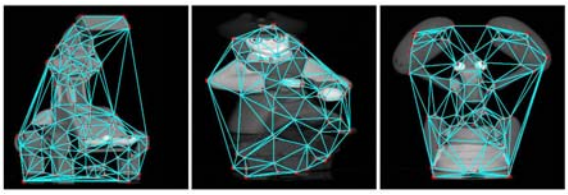


(d) Relative standard deviation.

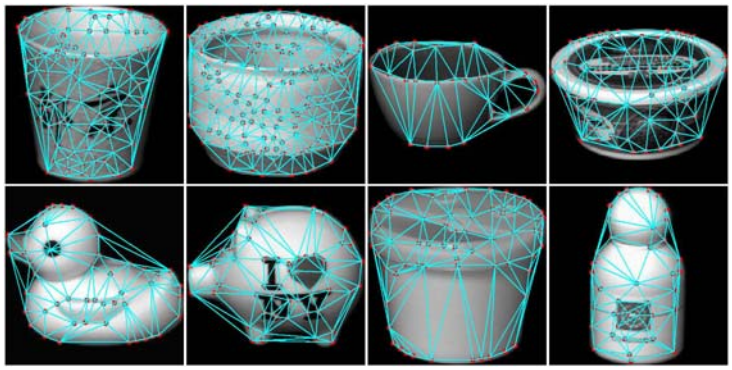
FIGURE 3.2: (A) (B) (C) REPRESENTATIONAL POWER ILLUSTRATED BY THE DISTANCE BETWEEN FEATURE VECTORS VERSUS GRAPH EDIT DISTANCE AND (D) STABILITY OF THE FEATURE DISTANCE TO PREDICT THE EDIT DISTANCE.



(a) House image samples.



(b) Toy image samples.



(c) Image samples in COIL dataset.

FIGURE 3.3: EXTRACTED GRAPHS FOR EXPERIMENTS.

interval [0,1], where 1 corresponds to a perfect clustering.

To take the study of establishing pattern vectors using Ihara coefficients one step further, we first explore which combination of the coefficients gives the best performance on the real world data because not all the coefficients are equally useful in characterizing graphs. This can be observed from Figure 3.4, where the pattern vector composed of a subset of the Ihara coefficients $\{c_3,\ c_4,\ c_5,\ c_6,\ c_7,\ \ln(|c_{2M}|)\}$ are separately applied to the house sequences and the first three objects in the COIL dataset. As the number of coefficients involved in a pattern vector increases, the Rand index does not always

exhibit a corresponding increase. This is in accordance with the experimental results on synthetic graphs in Section 3.4.1. Therefore, only a subset of the coefficients contribute significantly in representing graph features. That is, some coefficients may be redundant and some others may reduce the effectiveness of graph characterization. We thus need to select a subset of salient coefficients, i.e. those that take on distinct values for different classes and also exhibit small variance within the class. To do this, we compute the between-class scatter $S_b = \sum_{i=1}^{U} D_i(\bar{c}_{k,i} - \bar{c}_k)^2$ and the within-class scatter $S_w = \sum_{i=1}^{U} \sum_{c_{k,i,j} \in C_i} (c_{k,i,j} - \bar{c}_{k,i})^2$ of the individual coefficients, where $\bar{c}_k$ is the mean of the samples for coefficient c_k, $\bar{c}_{k,i}$ is the mean of the samples for c_k in class C_i, D_i is the number of samples in class C_i and U is the total number of classes. We then use the criterion function $J_C = (S_b + S_w)/S_w$ to evaluate the performance of the individual coefficients. We select the coefficients according to the condition that the individual coefficients that make the largest contributions to J_C are the most significant. We select three objects and for each object ten sample images are used as training data to compute the criterion function value. Figure 3.5 shows the criterion function values for the coefficients extracted from the three datasets. The beginning and trailing coefficients offer more discrimination than the intermediate ones. This is because the remaining coefficients provide no significant increase in information over c_3, c_4 and c_{2M} since they are determined by the number of triangles and squares in the graph and the vertex degrees, which are the most basic characteristics from md2 graphs. Based on this feature selection analysis, we work with the pattern vector $\vec{v}_{IS} = [c_3,\ c_4,\ \ln(|c_{2M}|)]^T$. The three components of $\vec{v}_{IS}$ extracted from the first four objects in the COIL dataset are shown in Figure 3.6 as a function of view number. Each line in a given plot represents the coefficients extracted from one object. The lines in each plot are well separated, thus indicating that the three coefficients are sufficient to distinguish different object classes.

We then apply the pattern vectors composed of a) truncated Laplacian spectra, i.e. the leading three nonzero Laplacian eigenvalues, b) unselected Ihara coefficients and c)

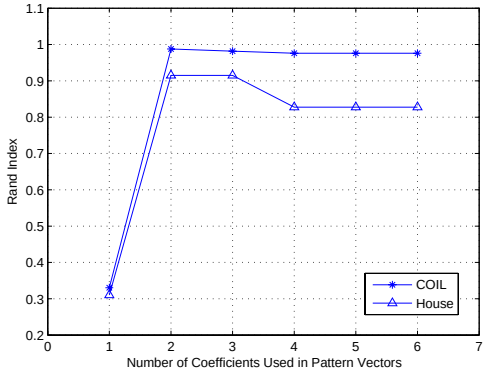


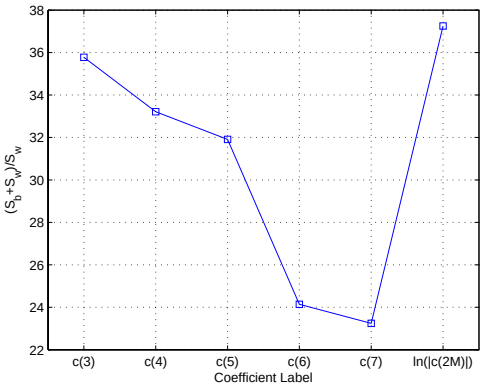
FIGURE 3.4: CLUSTERING PERFORMANCE FOR DIFFERENT NUMBERS OF COEFFICIENTS.

selected Ihara coefficients to the house sequences, toy pictures and the first four objects in the COIL dataset. Figures 3.7, 3.8 and 3.9 show the clustering results obtained by embedding the pattern vectors for the three datasets separately into a three-dimensional space using PCA. The selected coefficients outperform both the truncated Laplacian spectra and the unselected coefficients, and give clusters with better separation.

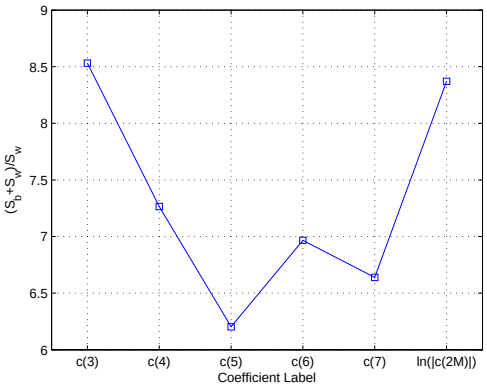
Finally we focus our attention on the COIL dataset, and evaluate the clustering performance obtained with different numbers of object classes. After embedding the pattern vectors into a three-dimensional space using PCA, we locate clusters using the K-means method and calculate the Rand index. The Rand indices for each kind of pattern vectors are shown in Table 3.1. The selected Ihara coefficients give the best performance and the truncated Laplacian spectra the poorest.

3.4.2 Edge-weighted Graphs

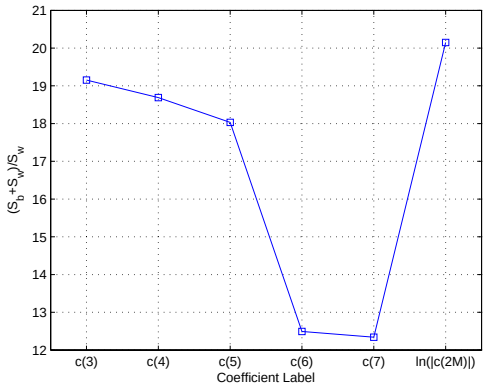
For edge-weighted graphs, we evaluate our proposed scheme in two ways. We again commence with a study on synthetic data aimed at evaluating the ability of the Ihara coefficients to distinguish between edge-weighted graphs under controlled structural errors.



(a) COIL data.



(b) Toy data.

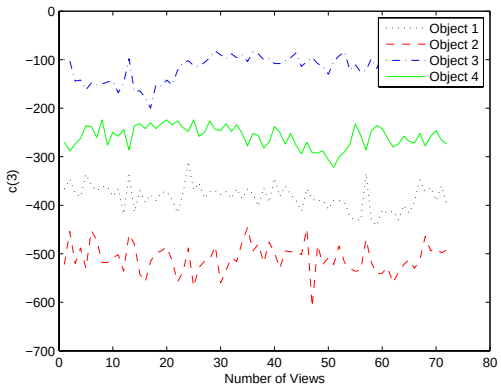


(c) House data.

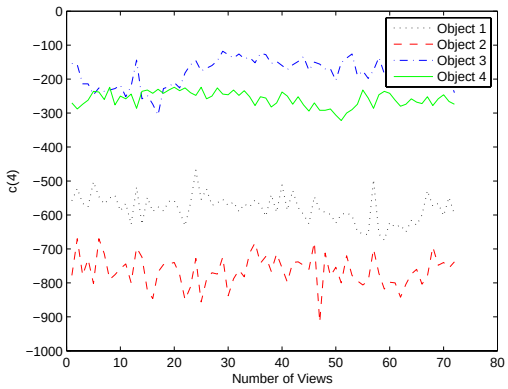
FIGURE 3.5: CRITERION FUNCTION VALUES FOR UNWEIGHTED GRAPHS.

Pattern vector	Number of object classes				
	4	5	6	7	8
Truncated Laplacian spectra	0.93	0.87	0.85	0.85	0.87
Unselected Ihara coefficients	0.98	0.88	0.88	0.85	0.86
Selected Ihara coefficients	0.99	0.93	0.87	0.87	0.88

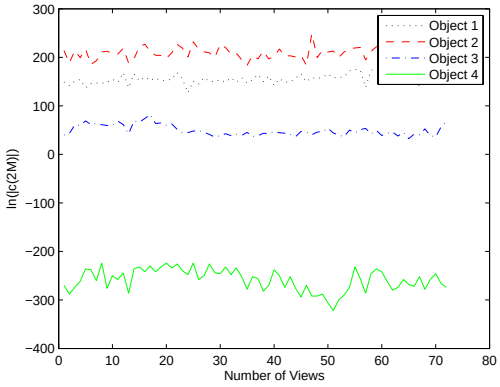
TABLE 3.1: RAND INDICES FOR UNWEIGHTED GRAPHS FROM THE COIL DATASET.



(a) c_3

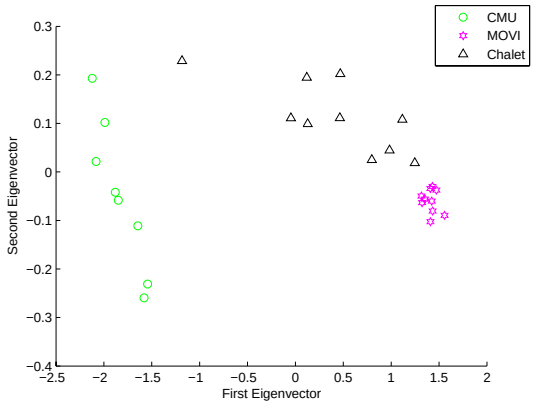


(b) c_4

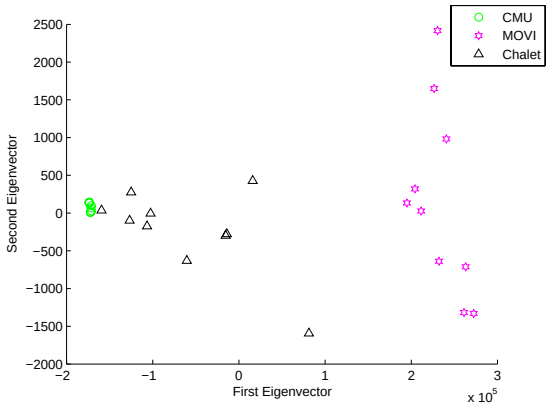


(c) $\ln(|c_{2M})|$

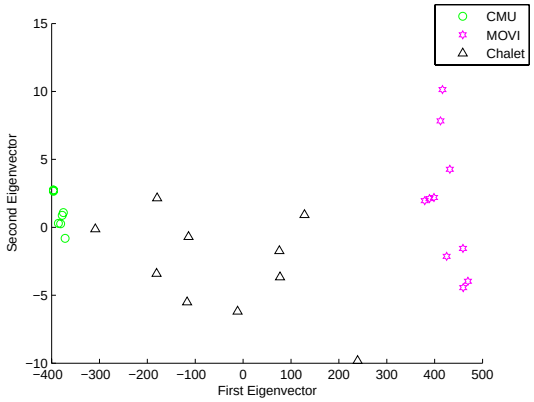
FIGURE 3.6: IHARA COEFFICIENTS FOR UNWEIGHTED GRAPHS EXTRACTED FROM THE COIL DATASET.



(a) Truncated Laplacian spectra for the house dataset.

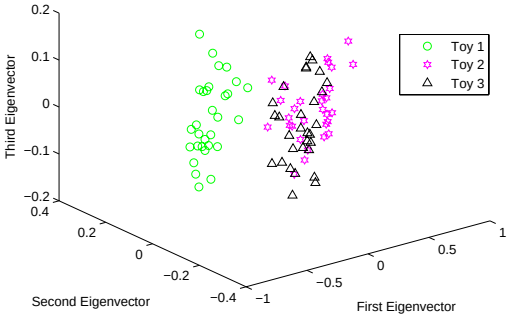


(b) Unselected Ihara coefficients for the house dataset.

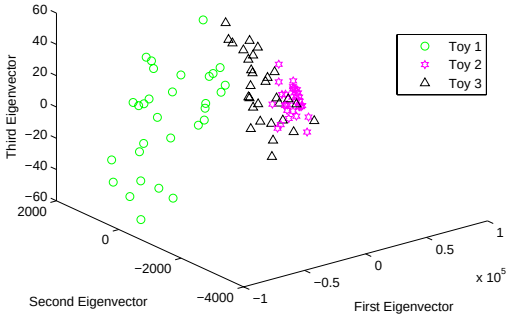


(c) Selected Ihara coefficients for the house dataset.

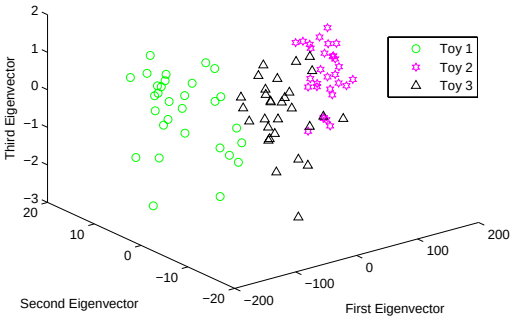
FIGURE 3.7: CLUSTERING PERFORMANCE FOR UNWEIGHTED GRAPHS EXTRACTED FROM THE HOUSE DATASET.



(a) Truncated Laplacian spectra for the toy data.

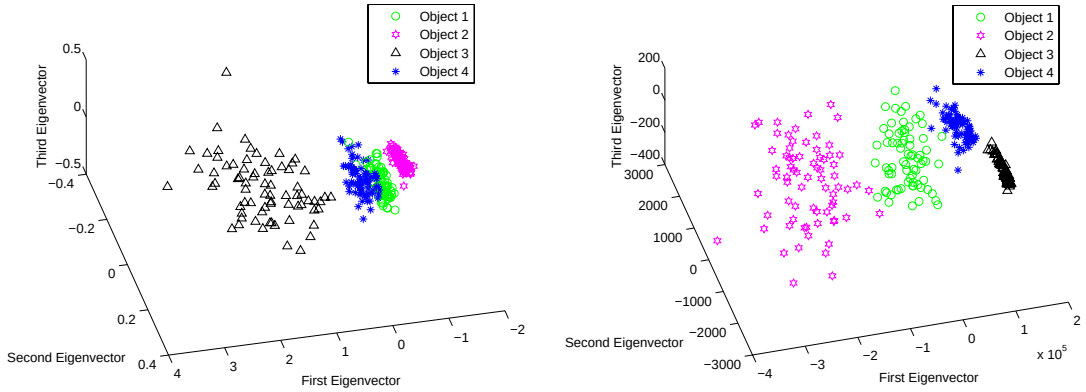


(b) Unselected Ihara coefficients for the toy data.



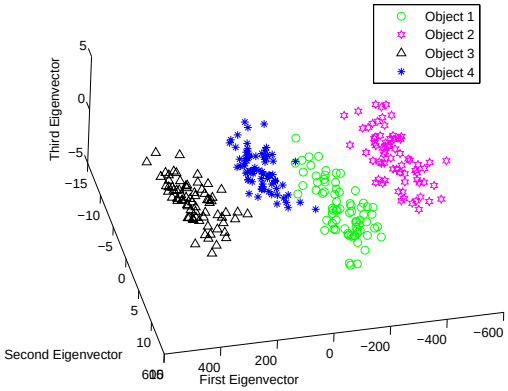
(c) Selected Ihara coefficients for the toy data.

FIGURE 3.8: CLUSTERING PERFORMANCE FOR UNWEIGHTED GRAPHS EXTRACTED FROM THE TOY DATASET.



(a) Truncated Laplacian spectra for the COIL dataset.

(b) Unselected Ihara coefficients for the COIL dataset.



(c) Selected Ihara coefficients for the COIL dataset.

FIGURE 3.9: CLUSTERING PERFORMANCE FOR UNWEIGHTED GRAPHS EXTRACTED FROM THE COIL DATASET.

Second, we focus on real world data and assess the effectiveness of the Ihara coefficient pattern vectors in detecting object clusters.

Synthetic Graphs

We first investigate the relationship between graph edit distance and the feature distance between Ihara coefficient pattern vectors for edge-weighted graphs. We commence with a single randomly generated md2 graph as the seed graph with 100 vertices and 300 weighted edges. In this subsection, the edge weights are always generated so as to have a uniform distribution over the interval [0.5, 1.5]. We obtain edited versions of the seed graph by randomly deleting edges, with the number of deleted edges varying from 1 to 30. For each number of deletions, we perform ten randomized edge deletion trials subject to the md2 constraint. We compute the Ihara coefficients using (3.13) and construct the pattern vector in the form of $\vec{v}_{WI} = [c_3,\, c_4,\, \ln(|c_{2M-3}|),\, \ln(|c_{2M-2}|),\, \ln(|c_{2M-1}|),\, \ln(|c_{2M}|)]^T$. The final four components of the pattern vector are scaled in a logarithmic manner to avoid unbalanced variance. Figure 3.10 plots the feature distances obtained using the pattern vectors composed of Ihara coefficients, versus the corresponding graph edit distances between the seed graph and its modified variants. The main feature to note from the plot is that for edge-weighted graphs the Ihara coefficient distance again generally follows the weighted edit distance. Moreover, for small distances the variation of Ihara coefficient distance is approximately linear with edit distance. For large edit distance the Ihara coefficient distance becomes more scattered.

To take this study on synthetic data one step further, we have studied the distribution of the Ihara coefficient feature distance. We investigate two sets of graphs. The first set consists of graphs which are obtained by randomly deleting one edge from a seed graph with 100 vertices and 301 weighted edges, subject to the md2 constraint. The second set are md2 graphs randomly generated with 100 vertices and 300 weighted edges. Figure 3.11(a) shows the distribution of the Euclidean feature distance between the vectors of

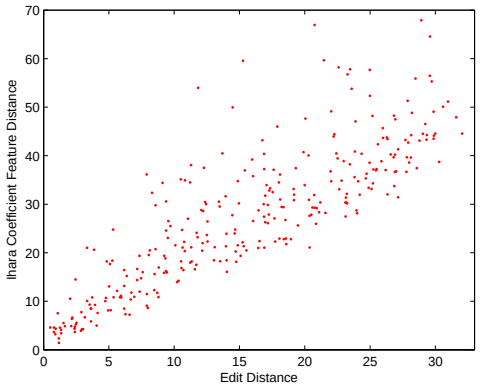
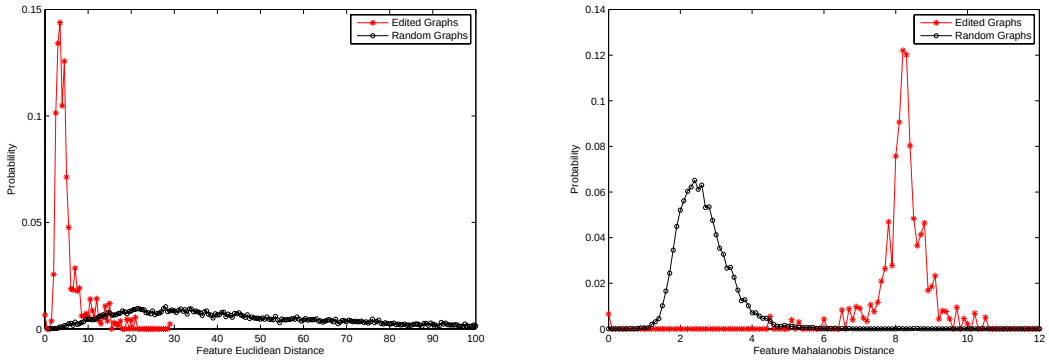
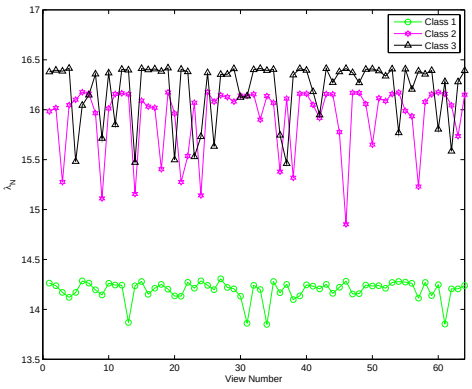


FIGURE 3.10: SCATTER PLOT OF THE DISTANCES COMPUTED USING THE IHARA COEFFICIENTS VERSUS THE CORRESPONDING EDIT DISTANCE.

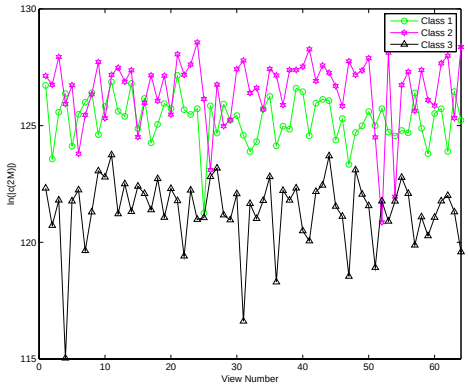


(a) Euclidean distance. (b) Mahalanobis distance.

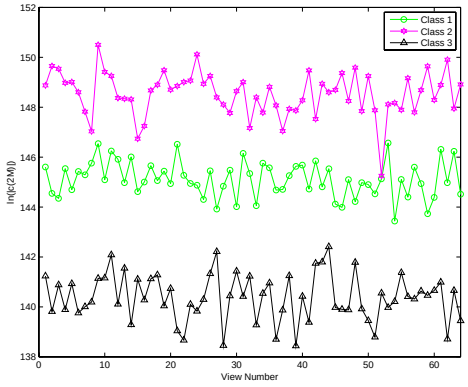
FIGURE 3.11: DISTANCE DISTRIBUTION OF RANDOM EDGE-WEIGHTED GRAPHS.



(a) Truncated Laplacian spectra.

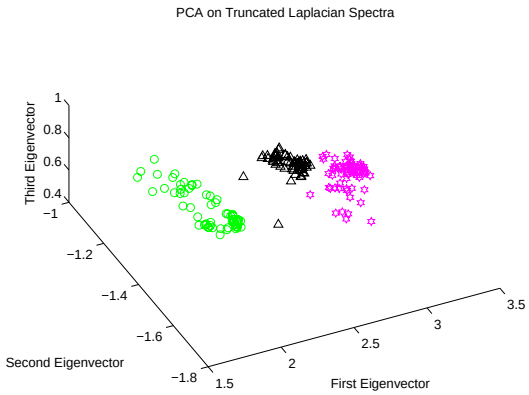


(b) ESP coefficients.

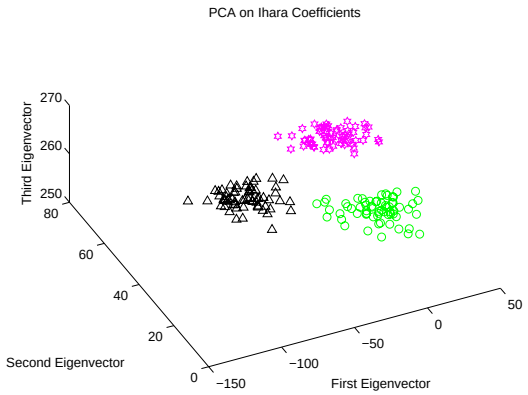


(c) Ihara coefficients.

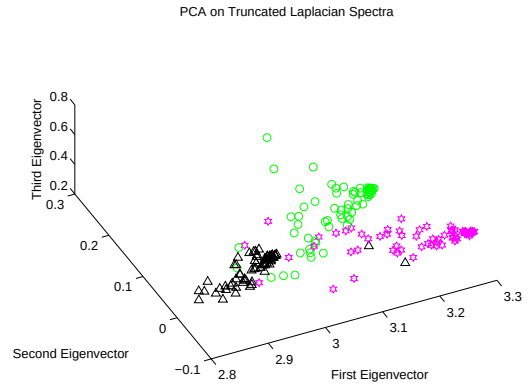
FIGURE 3.12: FEATURES FROM RANDOMLY GENERATED EDGE-WEIGHTED GRAPHS WITH A FIXED NUMBER OF EDIT OPERATIONS.



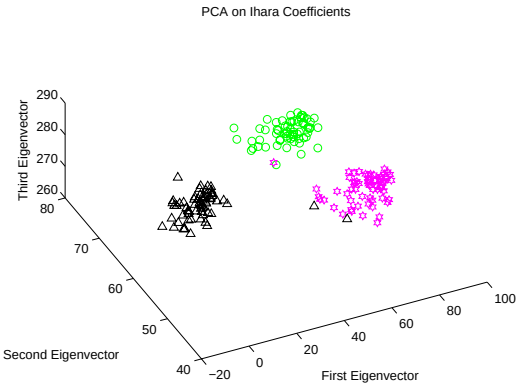
(a) Editing with a fixed number of vertex deletions.



(b) Editing with a fixed number of vertex deletions.



(c) Editing with a random number of vertex deletions.



(d) Editing with a random number of vertex deletions.

FIGURE 3.13: CLUSTERS FOR THREE CLASSES OF EDGE-WEIGHTED GRAPHS.

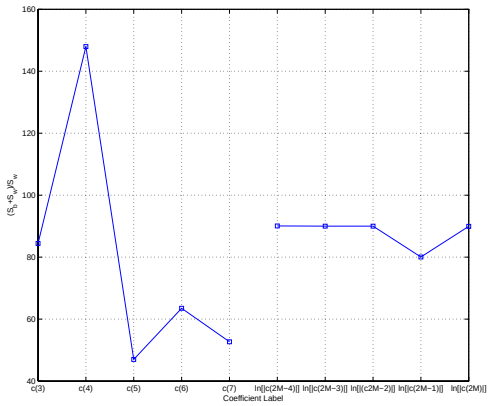
Ihara coefficients for the first set of graphs (red asterisk curve) and that of the second set (black circle curve). The modal distance between pattern vectors for graphs with random edge edits is much smaller than that for the structurally distinct graphs. For comparison, Figure 3.11(b) shows the distribution of the Mahalanobis distance for the two sets of graphs. The Mahalanobis distance between pattern vectors $\vec{v}_i$ and $\vec{v}_j$ is defined as $d_{i,j} = \sqrt{(\vec{v}_i - \vec{v}_j)^T \Sigma^{-1} (\vec{v}_i - \vec{v}_j)}$ where Σ is the covariance matrix of the pattern vectors. The Mahalanobis distance accounts for correlated features using the covariance and can correctly scale the feature components. The main feature to note in Figure 3.11 is that the two distributions are almost totally non-overlapping. From Figures 3.11(a) and 3.11(b), the distance between pattern vectors appears to provide scope for distinguishing between distinct graphs when there are variations in edge structure due to noise.

To provide an illustration and make a more comprehensive comparison with the graph spectral methods, we create two graph sets for testing the alternative representations. These two graph sets are established according to different types of graph edits separately. Each of the two sets are three classes of graphs separately derived from three seed graphs, which are again randomly generated with 100 vertices and 300 weighted edges. However, we perform two different types of edit operations on the seed graphs to establish the two graph sets separately. The first is to randomly delete eight edges at each time, and the second is with a random number of edge deletions from one to eight in each trial. We begin by performing the first type of edit operations on three seed graphs. Sixty four random trials of the edits are performed on each of the three seed graphs separately. Graph features of the three classes of edited graphs are shown in Figure 3.12, where Figures 3.12(a), 3.12(b) and 3.12(c) respectively show a) the largest Laplacian eigenvalue, b) the final coefficient of the elementary symmetric polynomial [102] of Laplacian spectrum (ESP's) and c) the final Ihara coefficient as a function of trial number. The main feature to note is that in the case of the Ihara coefficients, the variance is smallest and there is little overlap. The remaining two methods are overlapped to a more severe degree.

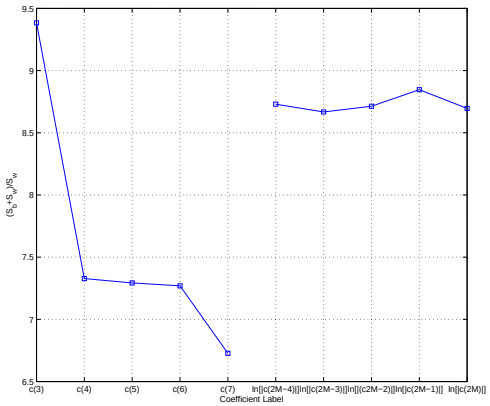
Next we embed the pattern vectors for the two sets of edited graphs separately into a three-dimensional space using PCA to evaluate their performance in clustering. We applied this procedure both to the Laplacian spectral pattern vector consisting of the second through to the seventh Laplacian eigenvalues and to the Ihara coefficient vector $\vec{v}_{WI}$. Figure 3.13 shows the experimental results. Figures 3.13(a) and 3.13(b) respectively show the clusters generated by the truncated Laplacian spectra and the Ihara coefficients, subject to the first type of graph edits. Although the Laplacian spectral method appears to produce 'good' clusters in Figure 3.13(a), there are some problems that are worth noting. First, the right two clusters are so close that they almost merge into one. Second, the left cluster has a non-compact ring shape and there are no graphs near the cluster center. However, the clusters in Figure 3.13(b) produced by the Ihara coefficient method are both more compact and more separable. Figures 3.13(c) and 3.13(d) show the results for the second type of graph edits. In this more complex situation, the Laplacian spectral method yields clusters with considerable scattering, as illustrated in Figure 3.13(c). However, for the Ihara coefficients, although there are a small number of outlier samples (two black triangles in the pink star cluster and one pink star in the green circle cluster), the overall performance is better and provides the basis for a usable clustering technique.

View-based Object Recognition

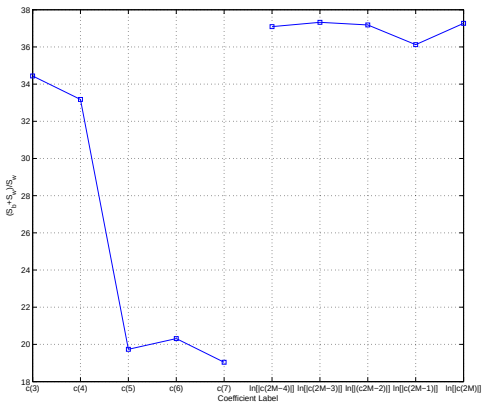
We now establish the pattern vectors composed of Ihara coefficients for edge-weighted graphs extracted from three datasets which have been used in Section 3.4.1, i.e. the house sequences, the toy pictures and the images from the COIL dataset. Here Delaunay graphs are also extracted as explained in Section 3.4.1. However, to consider edge-weighted graphs, the edge connecting vertices indexed by i and j is exponentially weighted by the negative of the Euclidean distance between the two vertices, i.e. $w_{ij} = \exp[-k\,||\boldsymbol{x}_i - \boldsymbol{x}_j||]$ where $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are coordinates of corner points i and j in an image and k is a scalar scaling factor.



(a) House data.

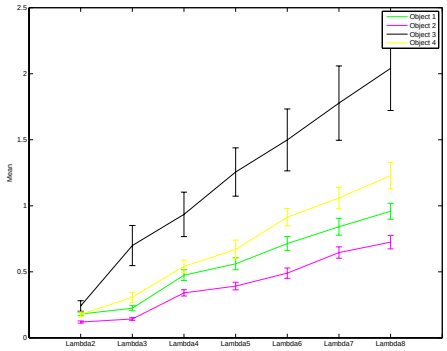


(b) Toy data.

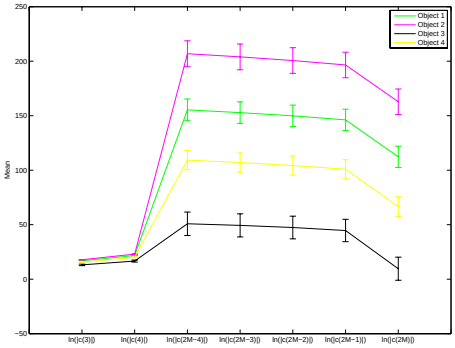


(c) COIL data.

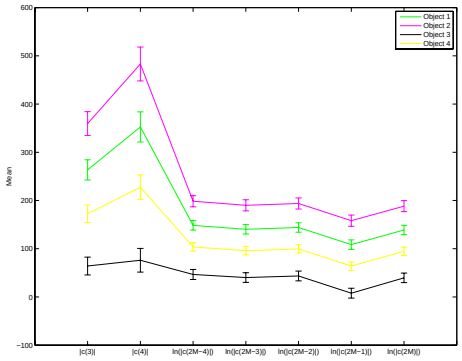
FIGURE 3.14: CRITERION FUNCTION VALUES FOR EDGE-WEIGHTED GRAPHS.



(a) Truncated Laplacian spectra.

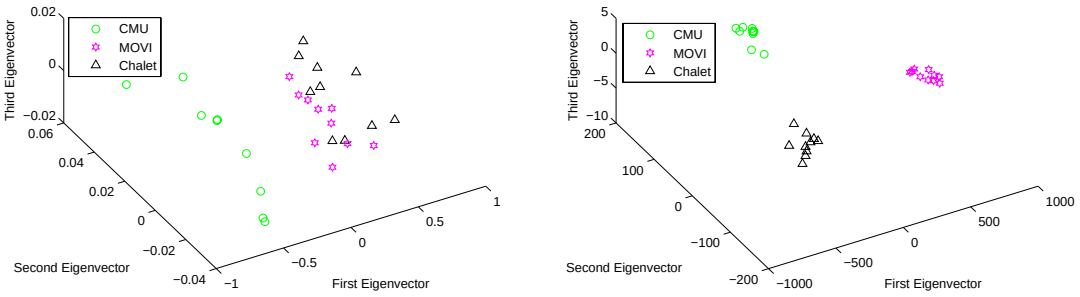


(b) ESP coefficients.



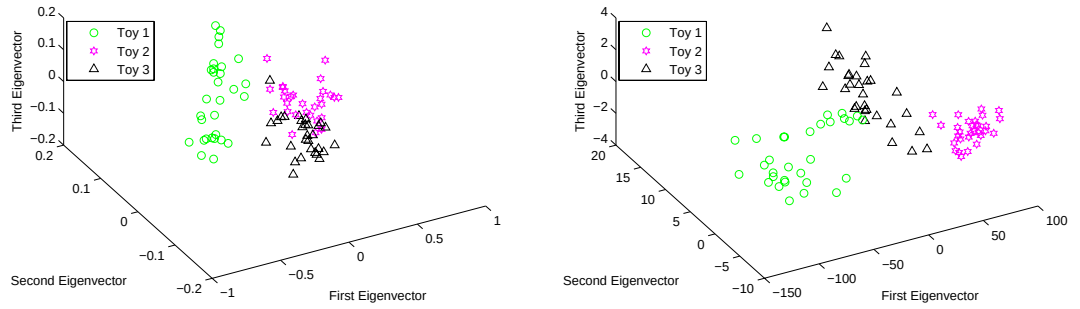
(c) Ihara coefficients.

FIGURE 3.15: STATISTICS OF THE FEATURES FROM EDGE-WEIGHTED GRAPHS EXTRACTED FROM THE COIL DATASET.



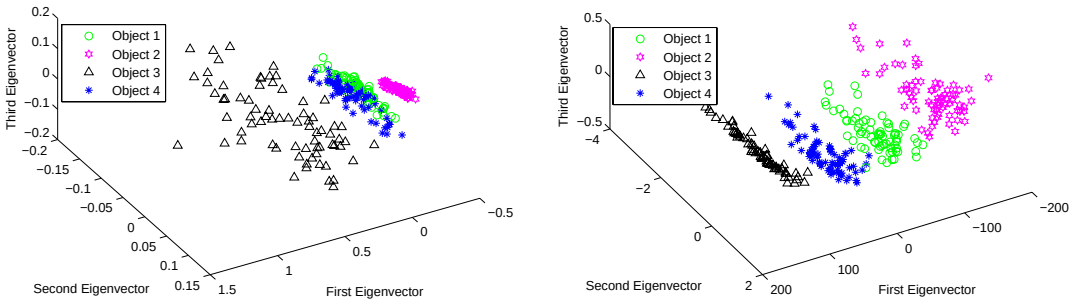
(a) Truncated Laplacian spectra for the house data. (b) Ihara coefficients for the house data.

FIGURE 3.16: CLUSTERING PERFORMANCE FOR EDGE-WEIGHTED GRAPHS EXTRACTED FROM THE HOUSE DATASET.



(a) Truncated Laplacian spectra for the toy data. (b) Ihara coefficients for the toy data.

FIGURE 3.17: CLUSTERING PERFORMANCE FOR EDGE-WEIGHTED GRAPHS EXTRACTED FROM THE TOY DATASET.



(a) Truncated Laplacian spectra for the COIL data. (b) Ihara coefficients for the COIL data.

FIGURE 3.18: CLUSTERING PERFORMANCE FOR EDGE-WEIGHTED GRAPHS EXTRACTED FROM THE COIL DATASET.

We first explore which coefficients provide the strongest discrimination between the graphs for the different object classes. To do this, we select the coefficients according to the criterion introduced in Section 3.4.1 that the individual coefficients which make the largest contributions to the criterion function are the most significant ones. For each of three selected objects, ten sample images are used as training data to compute the criterion function value. Figures 3.14(a), 3.14(b) and 3.14(c) show the criterion function values for the coefficients extracted from the house dataset, the toy dataset and the COIL dataset respectively. It is clear that the leading few and trailing coefficients contribute more to distinguishing the objects than the intermediate ones.

Next we evaluate the performance of the pattern vectors in distinguishing real world graph classes. We test the Laplacian eigenvalues, the ESP's and the Ihara coefficients on the COIL dataset. Figure 3.15 shows the statistics of the same four objects as a function of coefficient indices. For each coefficient we show the mean value and standard error over different views. For Figure 3.15, different colored lines correspond to different objects. The main features to note here are that a) both the Ihara coefficients and the ESP's are better separated than the Laplacian eigenvalues, b) there is now little difference between

the Ihara coefficients and the ESP's. This latter point is attributable to the high regularity of the Delaunay graphs.

We then use $\vec{v}_{WIS1} = [c_3,\ c_4,\ \ln(|c_{2M}|)]^T$, $\vec{v}_{WIS2} = [c_3,\ \ln(|c_{2M-1}|),\ \ln(|c_{2M}|)]^T$ and $\vec{v}_{WIS3} = [\ln(|c_{2M-2}|),\ \ln(|c_{2M-1}|),\ \ln(|c_{2M}|)]^T$ as the pattern vectors characterizing the edge-weighted graphs extracted from the house sequences, the toy pictures and the COIL dataset respectively. For comparison, we use the leading three non-zero Laplacian eigenvalues as the spectral pattern vector for the two datasets. We embed the pattern vectors into a three-dimensional space using PCA. Figures 3.16(a) and 3.16(b) show the clusters of graphs extracted from the house sequences, produced by the Laplacian spectral method and the Ihara coefficients respectively. Figures 3.17(a) and 3.17(b) show the clusters of graphs extracted from the three classes of toy pictures, produced by the Laplacian spectral method and the Ihara coefficients respectively. Figures 3.18(a) and 3.18(b) show the clusters of graphs extracted from the first four objects of the COIL dataset, produced by the Laplacian spectral method and the Ihara coefficients respectively. From Figures 3.16, 3.17 and 3.18 it is clear that the Ihara coefficients outperform the Laplacian spectral method in producing good clusters.

To take the quantitative evaluation of the pattern vectors one step further, we concentrate our attention on the COIL dataset, and evaluate the clustering performance obtained with different numbers of object classes. After performing PCA on the pattern vectors, we locate the clusters using the K-means method and calculate the Rand index for the resulting clusters. The Rand indices for the Laplacian spectral method and for the Ihara coefficients are listed in Table 3.2. From Table 3.2 it is clear that the Ihara coefficients outperform Laplacian spectra for all numbers of object classes studied.

Pattern vector	Number of object classes				
	4	5	6	7	8
Truncated Laplacian spectra	0.94	0.87	0.87	0.86	0.87
Ihara coefficients	0.99	0.95	0.90	0.88	0.89

TABLE 3.2: RAND INDICES FOR EDGE-WEIGHTED GRAPHS FROM THE COIL DATASET.

3.5 Summary

In this chapter we have studied how to extract characteristics from graphs using the Ihara zeta function, and have exploited the resulting characterization for the purposes of clustering graphs. We use Ihara coefficients to construct pattern vectors. Furthermore we have provided a route that allows the Ihara coefficients to be extended from unweighted to edge-weighted graphs. This is achieved by establishing the Perron-Frobenius operator for edge-weighted graphs with the assistance of a reduced Barthodi zeta function. This generalization allows us to compute the Ihara coefficients for both unweighted graphs and edge-weighted graphs in a unified manner. We have performed a spectral analysis that reveals the reasons why the Ihara coefficients are more effective in distinguishing graph classes than the graph spectral methods. Experiments have been conducted on both synthetic and real world data, and reveal not only that the Ihara coefficients are effective for graph clustering but that they also outperform the truncated Laplacian spectra.

Chapter 4

Hypergraph Characterization via Ihara Coefficients

In this chapter we present a hypergraph characterization method based on the Ihara coefficients. We verify that the Ihara coefficients are flexible for graphs and hypergraphs, and can distinguish structures consisting of vertices with the same pairwise connectivity but different relational orders. Furthermore, we present an efficient method for the coefficient computation. Experiments show both the effectiveness and efficiency of the proposed method.

4.1 Hypergraph Fundamentals

In this section, we review the definitions of hypergraphs and hypergraph Laplacian. Furthermore, we reveal the shortcoming of hypergraph Laplacian spectra in distinguishing hypergraphs when their differences are simply relationship orders, and explain the possible reasons for this deficiency.

4.1.1 Definition

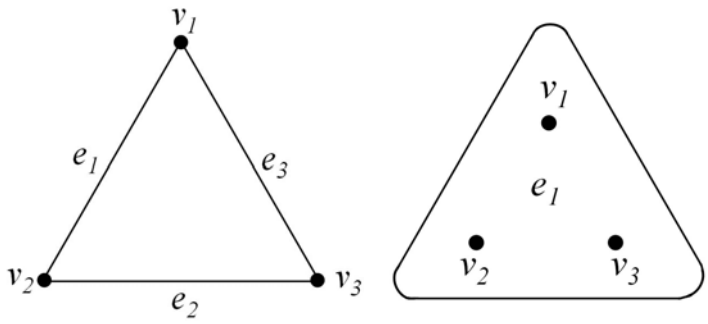
A hypergraph is defined as a set pair $HG(V, E)$ where V is a set of elements, called vertices, and E is a set of non-empty subsets of V called hyperedges. A hypergraph is a generalization of a graph. Unlike graph edges which are pairs of vertices, hyperedges are arbitrarily sized sets of vertices, and can therefore contain an arbitrary number of vertices. Examples of a hypergraph are shown in Figures 4.1 and 4.3. For the hypergraph in Figure 4.3, the vertex set is $V = \{v_1, v_2, v_3, v_4, v_5\}$ and the hyperedge set is $E = \{e_1 = \{v_1, v_3\},\ e_2 = \{v_1, v_2\},\ e_3 = \{v_2, v_4, v_5\},\ e_4 = \{v_3, v_4, v_5\}\}$. The representation of a hypergraph in the form of sets, concretely captures the relationship between vertices and hyperedges. However, it is difficult to manipulate this form in a computationally convenient way. Thus one alternative representation of a hypergraph is in the form of a matrix. For a hypergraph with I vertices and J hyperedges, we establish an $I \times J$ incidence matrix $\boldsymbol{H}$ with element $h_{i,j}$ as follows:

$$h_{i,j} = \begin{cases} 1 & \text{if } v_i \in e_j; \\ 0 & \text{otherwise.} \end{cases} \tag{4.1}$$

The incidence matrix of the hypergraph $HG(V, E)$ can be more easily manipulated than its equivalent set representation.

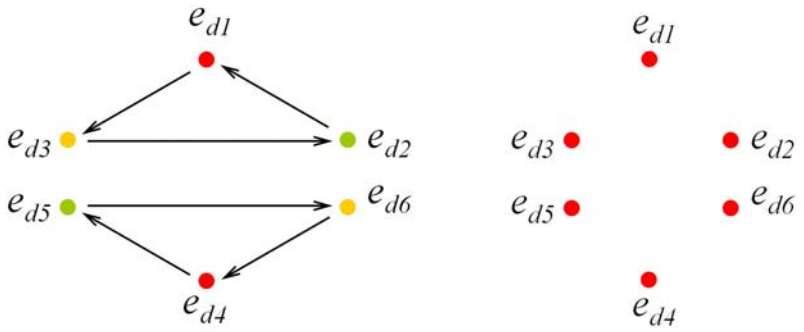
4.1.2 Hypergraph Laplacian Spectrum

Although the incidence matrix can fully describe the characteristics of a hypergraph, the matrix elements represent vertex-to-hyperedge relationships rather than vertex-to-vertex relationships. To obtain a vertex-to-vertex representation, we need to establish the adjacency matrix and Laplacian matrix for a hypergraph. To achieve this goal, a graph representation for the hypergraph is required. One possible method is to construct a graph with edges weighted by the quotient of the corresponding hyperedge weight and cardinality [112]. In this case, even those edges derived from an unweighted hyperedge are



(a) (b)

FIGURE 4.1: UNWEIGHTED HYPERGRAPHS WITH THE SAME ADJACENCY MATRIX AND LAPLACIAN MATRIX.



(a) Oriented line graph of Figure 4.1(a). (b) Oriented line graph of Figure 4.1(b).

FIGURE 4.2: ORIENTED LINE GRAPHS ASSOCIATED WITH THE HYPERGRAPHS IN FIGURE 4.1.

assigned a non-unit weight. As an alternative, rather than attaching a weight to each edge in the graph representation, one definition of the adjacency matrix and the associated Laplacian matrix for a nonuniform unweighted hypergraph are $\boldsymbol{A}_H = \boldsymbol{H}\boldsymbol{H}^T - \boldsymbol{D}_v$ and $\boldsymbol{L}_H = \boldsymbol{D}_v - \boldsymbol{A}_H = 2\boldsymbol{D}_v - \boldsymbol{H}\boldsymbol{H}^T$ respectively, where $\boldsymbol{D}_v$ is the diagonal vertex degree matrix whose diagonal element $d(v_i)$ is the summation of the elements in the ith row of $\boldsymbol{H}$ [67]. The set of eigenvalues of $\boldsymbol{L}_H$ are referred to as the hypergraph Laplacian spectrum and can be used in a straightforward way as characteristics captured from $HG(V, E)$.

However, these matrix representations exhibit deficiencies when used to characterize relational structures. For example, for the hypergraphs in Figures 4.1(a) and 4.1(b), the adjacency matrices are identical, and so are the associated Laplacian matrices. Specifically, the adjacency matrix and Laplacian matrix for the two hypergraphs in Figure 4.1 are as follows:

$$\boldsymbol{A}_H = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \qquad \boldsymbol{L}_H = \begin{pmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{pmatrix}.$$

It is clear that neither the unweighted adjacency matrix nor the Laplacian matrix can distinguish these two hypergraphs. One important reason for the limited usefulness of the above hypergraph matrix representations is that they result in information loss when relational orders of varying degree are present. The adjacency matrix and the Laplacian matrix introduced above only record the adjacency relationships between pairs of nodes and neglect the cardinalities of the hyperedges. In this regard they can not distinguish between pairwise relationships and higher order relationships for the same set of vertices. To overcome this deficiency, we must resort to tools that are capable of distinguishing hypergraphs with the same pairwise connectivity among the same set of vertices but with different relational orders. To this end, we adopt characteristic polynomials extracted from the Ihara zeta function as a means of representing hypergraphs. In the next section, we commence by showing that the Ihara zeta function can be used to represent this type

of relational structure in hypergraphs. We use the Ihara coefficients, i.e. the characteristic polynomial coefficients extracted from the determinant form of the Ihara zeta function, as hypergraph characteristics. We show that the Ihara coefficients not only encode the relational structure in a consistent way but also overcome the deficiencies caused by the vertex-to-vertex matrix representations described above.

4.2 Ihara zeta function from graphs to hypergraphs

The definition and rational expression of an Ihara zeta function for a graph have been introduced in Chapter 3. Based on the definition of the graph Ihara zeta function (3.1), Storm [91] has extended the definition of the Ihara zeta function to a hypergraph $HG(V, E)$ as follows:

$$\zeta_H(u) = \prod_{p \in P_H} \left(1 - u^{|p|}\right)^{-1}, \tag{4.2}$$

where P_H is the set of the equivalence classes of prime cycles in the hypergraph $HG(V, E)$. A prime cycle in a hypergraph is a closed path with no backtrack. That is, no hyperedge is traversed twice in the prime cycle. For example, the cycle $\{v_1 \to v_2 \to v_3 \to v_1\}$ in the hypergraph in Figure 4.1(a) is a prime cycle. However, the cycle $\{v_1 \to v_2 \to v_3 \to v_1\}$ in the hypergraph in Figure 4.1(b) is not a prime cycle. This is because the hyperedge e_1, e_2 and e_3 are only traversed once separately in the cycle $\{v_1 \to v_2 \to v_3 \to v_1\}$ in Figure 4.1(a). On the other hand, the cycle $\{v_1 \to v_2 \to v_3 \to v_1\}$ traverses the hyperedge e_1 more than once in Figure 4.1(b) and does not make it a prime cycle.

To formulate the Ihara zeta function for a hypergraph in a manner similar with (3.2), the bipartite graph representation of the hypergraph is needed. To establish the associated bipartite graph, we use a dual representation in which each hyperedge is represented by a new vertex. The union of the new vertex set and the original vertex set constitute the vertex set of the associated bipartite graph. Each new vertex is incident to each of original vertices in the corresponding hyperedge. The new vertices corresponding to hyperedges

are on one side and the original hypergraph vertices are on the other side. Thus the bipartite graph and star expansion for a hypergraph share the same form, although they are defined for different purposes. For instance, the bipartite graph associated with the example hypergraph in Figure 4.3 is shown in Figure 4.4.

The Ihara zeta function of the hypergraph $HG(V, E)$ can be equivalently expressed in a rational form as follows:

$$\zeta_H(u) = Z_{BG}(\sqrt{u}) = (1-u)^{\chi(BG)} \det\left(\boldsymbol{I}_{|V(HG)|+|E(HG)|} - \sqrt{u}\boldsymbol{A}_{BG} + u\boldsymbol{Q}_{BG}\right)^{-1}, \quad (4.3)$$

where $Z_{BG}(.)$ is the Ihara zeta function of the bipartite graph, $\chi(BG)$is the Euler number of the associated bipartite graph, $\boldsymbol{A}_{BG}$ is the adjacency matrix of the associated bipartite graph of $HG(V, E)$, and $\boldsymbol{Q}_{BG} = \boldsymbol{D}_{BG} - \boldsymbol{I}_{|V(HG)|+|E(HG)|}$. Further details on the arguments leading from (3.2) to (4.3) can be found in Storm's paper [91].

The adjacency matrix of the associated bipartite graph can be formulated using the incidence matrix $\boldsymbol{H}$ of $HG(V, E)$:

$$\boldsymbol{A}_{BG} = \begin{bmatrix} \boldsymbol{0}_{|E(H)|\times|E(H)|} & \boldsymbol{H}^T \\ \boldsymbol{H} & \boldsymbol{0}_{|V(H)|\times|V(H)|} \end{bmatrix}. \quad (4.4)$$

The hypergraph Ihara zeta function in the form of (4.3) provides an alternative method for the function value computation. It also results in a useful computational method which lifts the efficiency of computing the Ihara coefficients and will be discussed later on in Section 4.5.

4.3 Permutation Invariant

The structure of a hypergraph is invariant under permutations of nodes and hyperedge labels. If we want to represent a hypergraph using a pattern vector, we should ensure that the elements of the pattern vector are invariant to permutations of the vertex labels. However, the incidence matrix is modified by the vertex order and hyperedge order, since

the rows and columns are indexed by the vertex order and hyperedge order, respectively. If we relabel the vertices, the incidence matrix undergoes a permutation of rows, and if we relabel the hyperedges, the incidence matrix undergoes a permutation of columns. The elements of the adjacency matrix and Laplacian matrix for a hypergraph can be labeled by the vertex labels alone without taking into account the hyperedge labels. However, these two kinds of matrices also undergo column and row permutations when the vertex labels interchange. On the other hand, the hypergraph Ihara zeta function is invariant to label permutations on both vertices and hyperedges. To prove this property, suppose that $\boldsymbol{H}$ and $\boldsymbol{H}_P$ are the incidence matrices associated with two isomorphic hypergraphs $HG(V,E)$ and $HG_P(V,E)$, respectively. Let the matrix $\boldsymbol{P}_V$ be the node permutation matrix representing the change in node order between $HG(V,E)$ and $HG_P(V,E)$ and $\boldsymbol{P}_E$ the hyperedge permutation matrix representing the change in hyperedge order. Both $\boldsymbol{P}_V$ and $\boldsymbol{P}_E$ are orthogonal matrices and as a result $\boldsymbol{P}_V\boldsymbol{P}_V^T = \boldsymbol{I}_V$ and $\boldsymbol{P}_E\boldsymbol{P}_E^T = \boldsymbol{I}_E$, where $\boldsymbol{I}_V$ and $\boldsymbol{I}_E$ are identity matrices of the same size of $\boldsymbol{P}_V$ and $\boldsymbol{P}_E$, respectively. The permutation relationship between the two incidence matrices can be denoted as $\boldsymbol{H}_P = \boldsymbol{P}_V\boldsymbol{H}\boldsymbol{P}_E$. Thus, the relationship between the adjacency matrix of $HG_P(V,E)$ and that of $HG(V,E)$ is:

$$\boldsymbol{A}_{BG_P} = \begin{bmatrix} \boldsymbol{0}_{|E(HG)|\times|E(HG)|} & \boldsymbol{P}_E^T\boldsymbol{H}^T\boldsymbol{P}_V^T \\ \boldsymbol{P}_V\boldsymbol{H}\boldsymbol{P}_E & \boldsymbol{0}_{|V(HG)|\times|V(HG)|} \end{bmatrix} = \boldsymbol{P}\boldsymbol{A}_{BG}\boldsymbol{P}^T \tag{4.5}$$

where

$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{P}_E^T & \boldsymbol{0}_{|E(HG)|\times|V(HG)|} \\ \boldsymbol{0}_{|V(HG)|\times|E(HG)|} & \boldsymbol{P}_V \end{bmatrix} \tag{4.6}$$

and $\boldsymbol{P}$ is an orthogonal matrix with the size of $(|V(HG)| + |E(HG)|) \times (|V(HG)| + |E(HG)|)$.

Suppose that the Ihara zeta function for $HG_P(V,E)$ is:

$$\zeta_{H_P}(u) = (1-u)^{\chi(BG_P)}\det\left(\boldsymbol{I}_{|V(HG)|+|E(HG)|} - \sqrt{u}\boldsymbol{A}_{BG_P} + u\boldsymbol{Q}_{BG_P}\right)^{-1}. \tag{4.7}$$

Let $S = \det\left(\boldsymbol{I}_{|V(HG)|+|E(HG)|} - \sqrt{u}\boldsymbol{A}_{BG_P} + u\boldsymbol{Q}_{BG_P}\right)$, then

$$
\begin{aligned}
S &= \det\left(\boldsymbol{I}_{|V(HG)|+|E(HG)|} - \sqrt{u}\boldsymbol{P}\boldsymbol{A}_{BG}\boldsymbol{P}^T + u\boldsymbol{P}\boldsymbol{Q}_{BG}\boldsymbol{P}^T\right) \\
&= \det\left(\boldsymbol{P}\boldsymbol{I}_{|V(HG)|+|E(HG)|}\boldsymbol{P}^T - \sqrt{u}\boldsymbol{P}\boldsymbol{A}_{BG}\boldsymbol{P}^T + u\boldsymbol{P}\boldsymbol{Q}_{BG}\boldsymbol{P}^T\right) \\
&= \det\left[\boldsymbol{P}\left(\boldsymbol{I}_{|V(HG)|+|E(HG)|} - \sqrt{u}\boldsymbol{A}_{BG} + u\boldsymbol{Q}_{BG}\right)\boldsymbol{P}^T\right] \\
&= \det\left(\boldsymbol{I}_{|V(Hg)|+|E(HG)|} - \sqrt{u}\boldsymbol{A}_{BG} + u\boldsymbol{Q}_{BG}\right). \qquad (4.8)
\end{aligned}
$$

Thus we have

$$\zeta_{H_P}(u) = (1-u)^{\chi(BG_P)}S = \zeta_H(u). \qquad (4.9)$$

Therefore, the Ihara zeta function for a hypergraph and the characteristics derived from it are invariant to the permutations of both the vertex labels and hyperedge labels.

4.4 Determinant Expression of the Ihara Zeta Function for Hypergraphs

From the definition (4.2), it is clear that the Ihara zeta function for a hypergraph is the reciprocal of a polynomial:

$$\zeta_H(u) = (c_0 + c_1u + \cdots + c_{M-1}u^{M-1} + c_Mu^M)^{-1}, \qquad (4.10)$$

where M is the highest order of the polynomial, of which the detail will be discussed in Section 4.4.2. The polynomial coefficients $c_0, c_2, \ldots, c_M$ are referred to as the Ihara coefficients. Although the Ihara zeta function can be evaluated efficiently using (4.3), the task of enumerating the coefficients of the polynomial appearing in the denominator of the Ihara zeta function (4.3) is difficult, except by resorting to software for symbolic calculation. To efficiently compute these coefficients, a different strategy is adopted. The hypergraph is first transformed into an oriented line graph. The Ihara zeta function is then the reciprocal of the characteristic polynomial for the adjacency matrix of the oriented line graph.

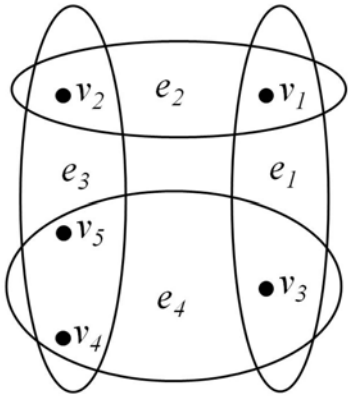


FIGURE 4.3: HYPERGRAPH EXAMPLE.

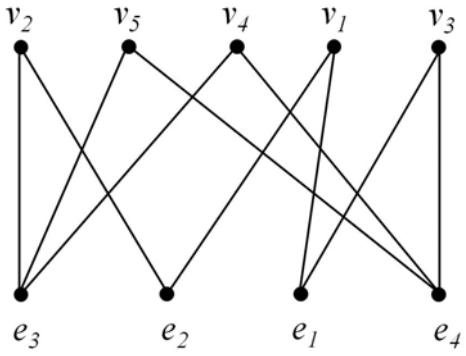


FIGURE 4.4: BIPARTITE GRAPH.

4.4.1 Oriented Line Graph

To establish the oriented line graph associated with $HG(V, E)$, we commence by constructing a $|e_i|$-clique, i.e. clique expansion, by connecting each pair of vertices in e_i through an edge for each hyperedge $e_i \in E$. The resulting graph is denoted by $GH(V, E_G)$. It is important to stress that there are potential multiple edges between two vertices in $GH(V, E_G)$ if the two vertices are encompassed by more than one common hyperedge in $HG(V, E)$. Suppose there are p hyperedges encompassing two vertices in $HG(V, E)$. The p hyperedges induces p edges separately between the two vertices in $GH(V, E_G)$. For the example hypergraph in Figure 4.3, $GH(V, E_G)$ is shown in Figure 4.5. In this example, the edges belonging to the common clique are indicated by the same color while the different cliques are colored differently. In the example, there are two edges between v_4 and v_5 colored differently, and these are induced by e_3 and e_4 in the original hypergraph in Figure 4.3, respectively.

For $GH(V, E_G)$, the associated symmetric digraph $DGH(V, E_d)$ can be obtained by replacing each edge of $GH(V, E_G)$ by an arc (oriented edge) pair in which the two arcs are inverse to one another. For $GH(V, E_G)$ in Figure 4.5, the associated $DGH(V, E_d)$ is

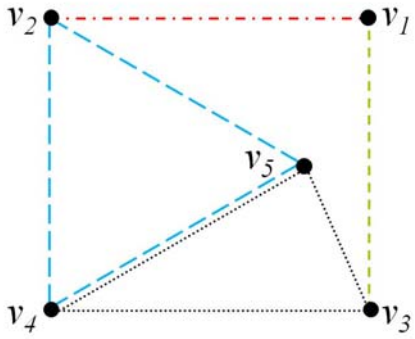


FIGURE 4.5: CLIQUE.

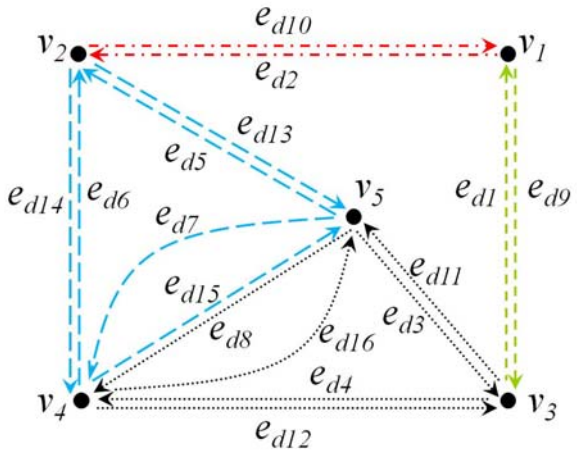


FIGURE 4.6: DI-CLIQUE.

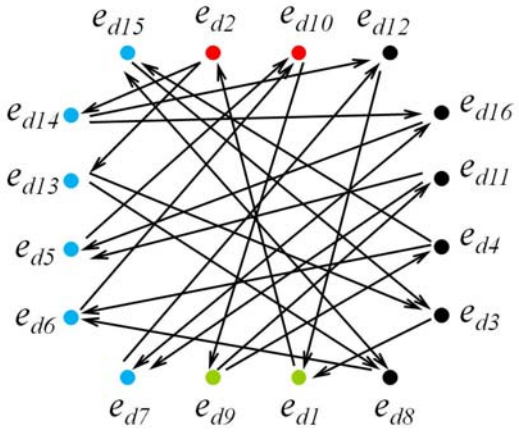


FIGURE 4.7: ORIENTED LINE GRAPH.

shown in Figure 4.6. Finally, the oriented line graph of the hypergraph can be established based on the symmetric digraph. The vertex set and edge set of the oriented line graph are defined as follows [91]:

$$\begin{cases} V_{ol} & = E_d; \\ E_{ol} & = \{(u,v)_i, (v,w)_j \in E_d \times E_d \,:\, i \neq j\}; \end{cases} \tag{4.11}$$

where the subscripts i and j denote the indices of the hyperedges from which the arcs (u, v) and (v, w) are, respectively, induced.

One observation that needs to be made here is that the adjacency matrix $\boldsymbol{A}_H$ and Laplacian matrix $\boldsymbol{L}_H$ for a hypergraph introduced in Section 4.1.2 are in fact those of the graph established on the clique expansion, but without an edge-weight attachment. These matrix representations can induce ambiguities when representing relational structures with different relational orders. This point is illustrated by the two hypergraphs in Figure 4.1, which have the same clique graph and thus the same adjacency matrix and Laplacian matrix. The reason for this is that the clique expansion only records adjacency relationships between pairs of vertices and can not distinguish whether or not two edges are derived from the same hyperedge. Thus the clique graph representations for a hypergraph may result in the loss of information concerning relational order. However, the Ihara zeta function overcomes this deficiency by avoiding the interaction between any two edges derived from the same hyperedge. This is due to the constraint in (4.11) that the connecting arc pair induced by the same hyperedge in the original hypergraph can not establish an oriented edge in the oriented line graph. On the other hand, if a pair of reverse arcs in $DGH(V, E_d)$ are induced by different hyperedges in the original hypergraph, they can establish a pair of reverse oriented edges in the oriented line graph. These properties are illustrated in Figure 4.7, which shows the oriented line graph obtained by transforming from the original hypergraph in Figure 4.3. From (4.11) it is clear that the vertices of the oriented line graph are obtained from the arcs of the symmetric digraph. As a result we can denote an arc in the digraph and its corresponding vertex in the oriented line graph

using the same label. In Figure 4.6 the terminus of the arc e_{d7} points to the origin of the arc e_{d6}. However, there is no oriented edge between vertices e_{d7} and e_{d6} in Figure 4.7. The reason for this is that they are derived from the same hyperedge e_5 in Figure 4.3. As a result, the constraint prevents connections between any nodes with the same color in Figure 4.7. On the other hand, there is a pair of reverse oriented edges (represented by one line with arrows on both ends) between e_{d15} and e_{d8} in Figure 4.7. The reason for this is that although e_{d15} and e_{d8} are reverse arcs in Figure 4.6, they are induced by different hyperedges, i.e. e_3 and e_4, respectively, in the original hypergraph in Figure 4.3. This is the also the case for e_{d7} and e_{d16} in Figures 4.6 and 4.7. As a result of these properties, the example hypergraphs in Figures 4.1(a) and 4.1(b), which have both the same adjacency matrix and the same Laplacian matrix, produce oriented line graphs with totally different structures as shown in Figures 4.2(a) and 4.2(b), respectively.

The adjacency matrix $\boldsymbol{T}_H$ of the oriented line graph is the Perron-Frobenius operator of the original hypergraph. For the (i, j)th entry of $\boldsymbol{T}_H$, $\boldsymbol{T}_H(i, j)$ is 1 if there is one edge directed from the vertex with label i to the vertex with label j in the oriented line graph, and otherwise it is 0. Unlike the adjacency matrix of an undirected graph, the Perron-Frobenius operator for a hypergraph is not a symmetric matrix. This is because the constraint in (4.11) arises in the construction of oriented edges. Specifically, any two arcs induced by the same hyperedge in the original hypergraph are not allowed to establish an oriented edge in the oriented line graph.

4.4.2 Characteristic Polynomial

With the oriented line graph to hand, the Ihara zeta function for a hypergraph can be written in the form of a determinant [91] using the Perron-Frobenius operator:

$$\zeta_H(u) = \det(\boldsymbol{I}_H - u\boldsymbol{T}_H)^{-1}. \tag{4.12}$$

From (4.12) we can see that M in (4.10) is the dimensionality of the square matrix

$\boldsymbol{T}_H$.

The Perron-Frobenius operators of the hypergraphs in Figures 4.1(a) and 4.1(b) are, respectively:

$$\boldsymbol{T}_{Ha} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}, \qquad \boldsymbol{T}_{Hb} = 0.$$

Substituting the above Perron-Frobenius operators into (4.12), we obtain the Ihara zeta functions for the hypergraphs in Figures 4.1(a) and 4.1(b) as follows:

$$\zeta_{Ha}(u) = (1 - 2u^3 + u^6)^{-1}; \tag{4.13}$$

$$\zeta_{Hb}(u) = 1. \tag{4.14}$$

The two functions have different forms and are thus available for distinguishing hypergraphs with the same clique expansion. This again verifies the validity of the Ihara zeta function in distinguishing hypergraphs which might cause ambiguities when traditional matrix representations such as adjacency matrix and Laplacian matrix are used.

From (4.12), we can see that the reciprocal Ihara zeta function for a hypergraph is in fact the characteristic polynomial of the Perron-Frobenius operator, that is

$$\zeta_H^{-1}(u) = \sum_{r=0}^{M} c_r u^r = \prod_{k=1}^{M} (u - \lambda_k) \tag{4.15}$$

where λ_k denotes the kth eigenvalue of the Perron-Frobenius operator $\boldsymbol{T}_H$.

To establish pattern vectors from the hypergraph Ihara zeta function for the purposes of characterizing hypergraphs in machine learning, it is natural to consider taking function samples as the elements. Although the function values at most of the sampling points will

perform well in distinguishing hypergraphs, there is the possibility of sampling at poles (when $u = \lambda_k$) giving rise to meaningless infinities. Hence, the pattern vectors consisting of function samples are potentially unstable representations of hypergraphs, since the distribution of the eigenvalues of $\boldsymbol{T}_H$ and hence the poles are unknown beforehand.

The characteristic polynomial coefficients in (4.10), i.e. the Ihara coefficients, do not give rise to infinities. Furthermore, these coefficients relate strongly to the hypergraph-structure since the Ihara zeta function records information about prime cycles in the hypergraphs. We use the Ihara coefficients as the elements of the pattern vector for a hypergraph and then apply them to clustering hypergraphs. From (4.15) it is clear that the characteristic polynomial coefficients are determined by the spectrum of the Perron-Frobenius operator. Each coefficient can be derived from the elementary symmetric polynomials of the eigenvalue set $\{\lambda_1, \lambda_2, \lambda_3 \ \ldots\}$ of $\boldsymbol{T}_H$ as follows:

$$c_r = (-1)^r \sum_{k_1<k_2< \ \ldots \ <k_r} \lambda_{k_1}\lambda_{k_2} \ \ldots \ \lambda_{k_r}. \tag{4.16}$$

4.5 Numerical Computation

We can compute the Ihara zeta function values by using (4.3). However, to efficiently compute the Ihara coefficients, we need to compute the characteristic polynomial of $\boldsymbol{T}_H$ in (4.12) and thus obtain the function form in (4.10) to compute the coefficient set. However, the construction of the oriented line graph for a hypergraph is complicated and computationally expensive. Furthermore, the eigencomputation on $\boldsymbol{T}_H$ tends to induce large values which often leads to floating point overflow errors. Therefore, the computation of the Ihara coefficients from the characteristic polynomial of $\boldsymbol{T}_H$ is impractical in real-world situations.

To overcome the deficiency of computing the Ihara coefficients using (4.12) and (4.16), in this section, we develop a novel yet straightforward method which commences from the associated bipartite graph. Instead of constructing the oriented line graph for a hy-

pergraph, we establish the oriented line graph for the bipartite graph associated with the hypergraph. The construction of the oriented line graph for a graph (at this stage for the associated bipartite graph) is simply a special case of the procedures described in Section 4.4.1 as long as the hypergraph is 2-uniform. Considering the rational expression (4.3) based on the associated bipartite graph, we have

$$\zeta_H^{-1}(u) = Z_{BG}^{-1}(\sqrt{u}) = \det(\boldsymbol{I}_{BG} - \sqrt{u}\boldsymbol{T}_{BG}) \tag{4.17}$$

where $\boldsymbol{T}_{BG}$ is the Perron-Frobenius operator of the associated bipartite graph, of which the reciprocal Ihara zeta function is represented as

$$\begin{aligned} Z_{BG}^{-1}(u) &= \prod_{p \in P_{BG}} \left(1 - u^{|p|}\right) \\ &= \left(1 - u^{|p_1|}\right)\left(1 - u^{|p_2|}\right)\left(1 - u^{|p_3|}\right)\cdots, \end{aligned} \tag{4.18}$$

where p_i is the ith prime cycle in the set P_{BG} of prime cycle equivalence classes of the bipartite graph. Note that every prime cycle in a bipartite graph has an even length, i.e. $|p_i|$ is always an even number for a bipartite graph. Let $\{\tilde{c}_0, \tilde{c}_1, \tilde{c}_2, \tilde{c}_3, \tilde{c}_4, \tilde{c}_5, \tilde{c}_6 \ldots\}$ denote the Ihara coefficient set for the bipartite graph. It is clear that $Z_{BG}^{-1}(u)$ is a polynomial with the odd coefficients equal to zero, that is

$$\begin{aligned} Z_{BG}^{-1}(u) &= \det(\boldsymbol{I}_{BG} - u\boldsymbol{T}_{BG}) \\ &= \tilde{c}_0 + \tilde{c}_1 u + \tilde{c}_2 u^2 + \tilde{c}_3 u^3 + \tilde{c}_4 u^4 + \tilde{c}_5 u^5 + \tilde{c}_6 u^6 + \cdots \\ &= \tilde{c}_0 + 0u + \tilde{c}_2 u^2 + 0u^3 + \tilde{c}_4 u^4 + 0u^5 + \tilde{c}_6 u^6 + \cdots \\ &= \tilde{c}_0 + \tilde{c}_2 u^2 + \tilde{c}_4 u^4 + \tilde{c}_6 u^6 + \cdots. \end{aligned} \tag{4.19}$$

Therefore, if we take $\sqrt{u}$ as the argument of the Ihara zeta function of the bipartite graph

instead of u, we have

$$\begin{aligned}
\zeta_H^{-1}(u) &= Z_{BG}^{-1}(\sqrt{u}) \\
&= \det(\boldsymbol{I}_{BG} - \sqrt{u}\boldsymbol{T}_{BG}) \\
&= \left(1-(\sqrt{u})^{|p_1|}\right)\left(1-(\sqrt{u})^{|p_2|}\right)\left(1-(\sqrt{u})^{|p_3|}\right)\cdots \\
&= \tilde{c}_0 + 0\sqrt{u} + \tilde{c}_2(\sqrt{u})^2 + 0(\sqrt{u})^3 + \tilde{c}_4(\sqrt{u})^4 + 0(\sqrt{u})^5 + \tilde{c}_6(\sqrt{u})^6 + \cdots \\
&= \tilde{c}_0 + \tilde{c}_2 u + \tilde{c}_4 u^2 + \tilde{c}_6 u^3 + \cdots \\
&= c_0 + c_1 u + c_2 u^2 + c_3 u^3 + \cdots . \qquad (4.20)
\end{aligned}$$

As we can see from (4.20), the polynomial coefficients of the hypergraph can be efficiently obtained by selecting just the even-indexed Ihara coefficients of the associated bipartite graph. Suppose $\{\tilde{\lambda}_1, \tilde{\lambda}_2, \tilde{\lambda}_3 \ ... \ \}$ is the eigenvalue set associated with $\boldsymbol{T}_{BG}$, then each Ihara coefficient can be computed as

$$c_r = \tilde{c}_{2r} = \sum_{k_1<k_2< \ ... \ <k_{2r-1}<k_{2r}} \tilde{\lambda}_{k_1}\tilde{\lambda}_{k_2} \ ... \ \tilde{\lambda}_{k_{2r-1}}\tilde{\lambda}_{k_{2r}}. \qquad (4.21)$$

The high computational overheads associated with $\boldsymbol{T}_H$ are due to the large size of the matrix $\boldsymbol{T}_H$. On the other hand, $\boldsymbol{T}_{BG}$ overcomes this drawback since it is much smaller in size than $\boldsymbol{T}_H$, especially for hypergraphs with large hyperedge cardinalities. The size of the Perron-Frobenius operator of a nonuniform hypergraph tends to be difficult to enumerate. Here we thus use the K-uniform hypergraph, i.e. hypergraph with every hyperedge containing K vertices, for analyzing the computational complexity of the Perron-Frobenius operators $\boldsymbol{T}_H$ and $\boldsymbol{T}_{BG}$. Suppose there are in total N hyperedges in the K-uniform hypergraph. To obtain $\boldsymbol{T}_H$, the clique expansion and its digraph of the K-uniform hypergraph need to be established according to the transform introduced in Section 4.4.1. This procedure produces an oriented line graph with $K(K-1)N$ vertices and a Perron-Frobenius operator of size $(K-1)KN \times (K-1)KN$. To obtain $\boldsymbol{T}_{BG}$, the bipartite graph and its digraph of the K-uniform hypergraph need to be established. This

procedure produces an oriented line graph with $2KN$ vertices and a Perron-Frobenius operator of size $2KN \times 2KN$. For uniform hypergraphs K is greater than 2, and the relation always holds for $2KN \leq (K-1)KN$. As a result, the size of $\boldsymbol{T}_{BG}$ is smaller than that of $\boldsymbol{T}_H$ as long as $K > 3$. Additionally, the computational complexity of obtaining the Ihara coefficients is governed by the eigendecomposition of the Perron-Frobenius operator. This requires $O(n^3)$ operations where n is the size of the Perron-Frobenius operator. Therefore, the computational overheads of eigendecomposition on $\boldsymbol{T}_{BG}$ are lower than those of $\boldsymbol{T}_H$.

4.6 Experiments

Our experiments are aimed at evaluating both the effectiveness and efficiency of our proposed method. In Section 4.6.1, we test our proposed method on real-world data with the aim of assessing the representational power of the Ihara coefficients for hypergraph characterization. In Section 4.6.2, we measure the computational overheads of our numerical method for computing the hypergraph Ihara coefficients (presented in Section 4.5) and compare it with the original method based on the Perron-Frobenius operator $\boldsymbol{T}_H$.

4.6.1 Hypergraph Clustering

In this subsection, we evaluate the effectiveness of the Ihara coefficients in clustering hypergraphs. Specifically, we assess whether the hypergraph Ihara coefficients can be embedded into a pattern space that reveals their cluster structure. We also compare the method based on hypergraph Ihara coefficients with the spectral characterization of hypergraphs.

This subsection is divided into two parts. Firstly, we introduce a hypergraph extraction method based on high-level feature points of images. Secondly we test our proposed hypergraph characterization method on hypergraphs extracted from 2D different views of

3D objects. Both qualitative and quantitative evaluations are made to verify the effectiveness of the proposed method.

Hypergraph Representation of Objects

To the best of our knowledge, Bretto et al. [11] were among the first to propose the use of hypergraph-based image representations. In their work, they incorporate the intensity values of picture elements into the neighborhood description for edge detection and denoising. However, this representation is low level pixel-based and is computationally burdensome.

Here we adopt a high-level hypergraph representation, i.e. feature hypergraph, based on feature points, rather than the pixel contents of an image. The hypergraph associated with one object is established with m vertices and m hyperedges, where m is the cardinality of the feature point set $V = \{v_1, v_2 \dots v_m\}$ extracted from the object. Each feature point is represented by one vertex in the hypergraph and therefore the feature point set is identical to the hypergraph vertex set (denoted by V). Each hyperedge is specific to one vertex, which we define as the seed of the hyperedge.

To determine the elements of each hyperedge in addition to the seed, we measure the similarities between the intensities of the seed and of the other vertices which lie in the spatial neighborhood of the seed. This confines the extent of the hyperedge. Let $\boldsymbol{c}(v_i)$ denote the spatial coordinate of the feature point v_i in an image, $I(v_i)$ denote the intensity of v_i, Φ_{j1} be a fixed value which represents the distance threshold for neighborhood, and Φ_{j2} be the similarity threshold between the vertex intensities. The similarity threshold Φ_{j2} could be either a fixed value or calculated in an adaptive manner. In our experiments, we adapt the threshold Φ_{j2} so that it is equal to the standard deviation of the intensities of the neighboring feature point

$$\Phi_{j2} = \frac{1}{N_{Neigh} - 1} \sqrt{\sum_{i,\ \|\boldsymbol{c}(v_i) - \boldsymbol{c}(v_j)\| \leq \Phi_{j1}} [I(v_i) - I(v_j)]^2} \tag{4.22}$$

(a) Houses.

(b) Toys.

(c) COIL.

FIGURE 4.8: DATASETS.

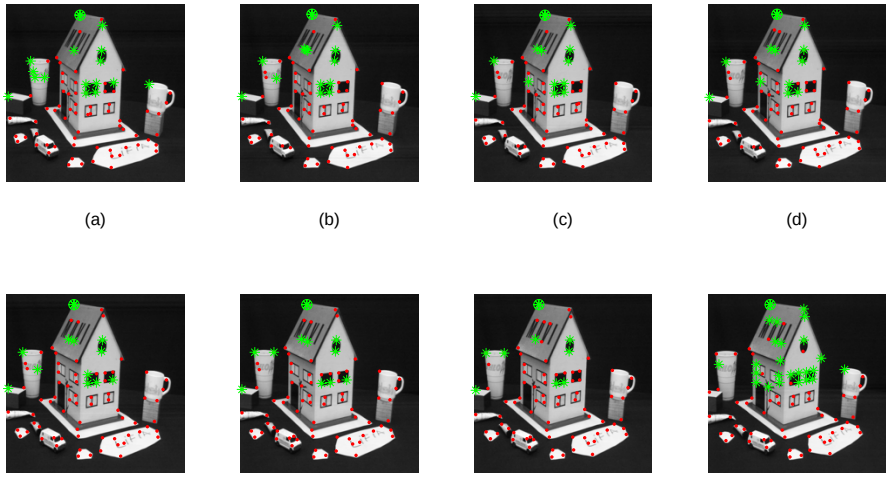


FIGURE 4.9: HYPEREDGES ENCOMPASSING THE SELECTED FEATURE POINT IN THE MOVI HOUSE IMAGES.

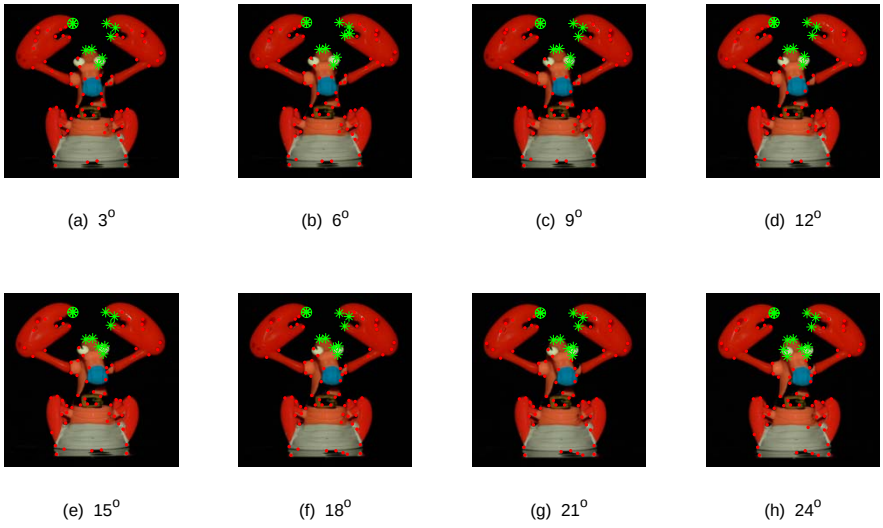


FIGURE 4.10: HYPEREDGES ENCOMPASSING THE SELECTED FEATURE POINT IN THE TOY LOBSTER IMAGES.

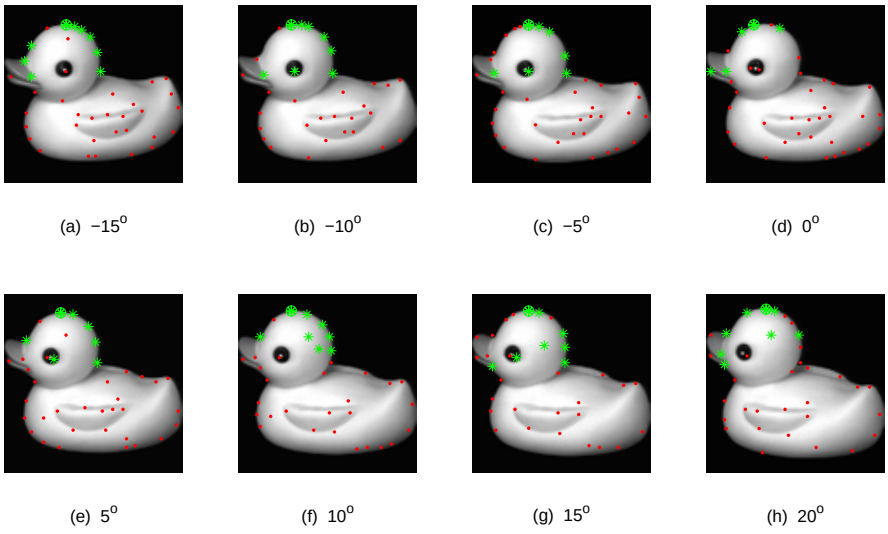


FIGURE 4.11: HYPEREDGES ENCOMPASSING THE SELECTED FEATURE POINT IN THE TOY DUCK IMAGES.

where N_{Neigh} is the number of feature points in the neighborhood of the feature point v_j.

The association of the vertex v_i to the hyperedge $E(v_j)$ derived from the seed v_j is determined as follows

$$v_i \begin{cases} \in E(v_j) & \text{if } \|\boldsymbol{c}(v_i) - \boldsymbol{c}(v_j)\| \leq \Phi_{j1} \text{ and if } \mid I(v_i) - I(v_j) \mid \leq \Phi_{j2}; \\ \notin E(v_j) & \text{otherwise.} \end{cases} \tag{4.23}$$

The first condition confines one set of candidate vertices to the spatial neighborhood of the seed. The second condition selects a set of candidate vertices satisfying the similarity constraint. The intersection of the two sets constitute one hyperedge. We refer to the hypergraph established in (4.23) as the feature hypergraph of an object.

The incidence matrix of the feature hypergraph is hence of size $m \times m$. The ith column represents the hyperedge derived from v_i, and the jth row corresponds to vertex v_j. Thus

the element of incidence matrix is denoted as

$$h(i,j) = \begin{cases} 1 & \text{if } \|\boldsymbol{c}(v_i) - \boldsymbol{c}(v_j)\| \leq \Phi_{j1} \text{ and if } \mid I(v_i) - I(v_j) \mid \leq \Phi_{j2}; \\ 0 & \text{otherwise.} \end{cases} \tag{4.24}$$

This strategy for extracting hypergraphs from images is similar to the method introduced in [11], where the common feature of these two methods is that the adaptive similarity threshold is used to capture the local variance. The main difference between them is that the hypergraphs established in [11] are based on all the pixels in an image and is a low-level visual representation. The hypergraphs thus obtained tend to be large in size and induce expensive computational overheads. Our method, on the other hand, is based on feature points rather than the complete image pixel set and is a high-level representation. We establish hypergraphs with reasonably small sizes, which are easy to compute and process.

In our experiments on real world data, we extract hypergraphs from 2D image sequences of 3D objects. The first set of hypergraphs are extracted from house images in the CMU, MOVI and Chalet sequences (samples are shown in Figure 4.8(a)), the second set are extracted from images of three toys (samples are shown in Figure 4.8(b)), and the third set are extracted from images of eight objects in the COIL dataset (samples are shown in Figure 4.8(c)). In these datasets, the pictures for each object are taken from different view angles. There are ten pictures taken for each model house, thirty pictures for each toy, and seventy two pictures for each object in the COIL database. To establish hypergraphs on the objects, we first extract feature points using the Harris detector [39] as the vertices of hypergraphs. For each image, we construct the feature hypergraph using the method presented in this subsection. Here the neighborhood threshold Φ_{j1} is set to be 1/4 the size of the image, and similarity threshold Φ_{j2} is computed using (4.22). We visualize one object of each dataset by sequencing eight of its consecutive images in Figures 4.9, 4.10 and 4.11 separately. The eight views of the toy lobster in Figure 4.10 are taken consecutively with interval 3°. The eight views of the toy duck in Figure 4.11 are taken

consecutively with interval 5°. In Figures 4.9, 4.10 and 4.11, the feature points extracted by the Harris detector are super-imposed on the images. Here we highlight an example seed vertex as a circled green asterisk. Those vertices belonging to the same hyperedge spanned by the seed vertex are shown as uncircled green asterisks. The remaining feature points are shown as red dots. From the examples in the different sequences it is clear that the hyperedge is generally stable under the change of viewpoint.

Hypergraph Ihara Coefficients for Clustering

We evaluate the effectiveness of the Ihara coefficients in clustering hypergraphs extracted from the different image sequences in Section 4.6.1. We use this problem as an application vehicle to demonstrate the utility of our hypergraph characterization. It must be stressed that our encoding of image feature arrangements does not necessarily represent the optimal way to use this information, and much research remains to be done to define how the information can be used in an optimal way. Moreover, the imagery used in our study does not necessarily provide a demanding test of the method. However, the data has been used in a number of earlier studies [16][103] of graph-based method for object view clustering. In fact the object recognition problem posed by the imagery is quite straightforward and could probably be solved using very simple image statistics such as gray-scale or color histograms. Nonetheless, it must be stressed that our representation based on the Ihara zeta function provides a characterization of the arrangement of image features that is invariant to translation, rotation and scaling. We stress that the literature on image representation using hypergraphs is still in its infancy and little is available as a standard of comparison or benchmark.

We commence by exploring which coefficients provide the strongest discrimination between the hypergraphs for the different object classes. To do this, we compute the between-class scatter $S_b = \sum_{i=1}^{U} D_i(\bar{c}_{k,i} - \bar{c}_k)^2$ and the within-class scatter $S_w = \sum_{i=1}^{U} \sum_{c_{k,i,j} \in C_i} (c_{k,i,j} - \bar{c}_{k,i})^2$ of the individual coefficients, where $\bar{c}_k$ is the mean of the samples

for coefficient c_k, $\bar{c}_{k,i}$ is the mean of the samples for c_k in class C_i, D_i is the number of samples in class C_i and U is the total number of classes. We then use the criterion function $J_C = (S_b + S_w)/S_w$ to evaluate the performance of the individual coefficients. Figure 4.12 illustrates the function values of J_C for the beginning and trailing hypergraph Ihara coefficients for the house dataset. For the trailing coefficients, the criterion function values for both the original coefficients and the natural logarithm scaled coefficients are plotted. It is clear that the natural logarithm scaled trailing coefficients are more distinctive than the original trailing coefficients and the leading ones, and there is a tendency that the leading coefficients are more discriminative than the intermediate ones. Figure 4.13 shows the logarithm of the coefficient standard deviation as a function of coefficient index. It is clear that the standard deviation of the trailing coefficients is many orders of magnitude larger than that of the leading coefficients. It is for this reason that we re-scale the trailing coefficients by taking the natural logarithm in our experiments.

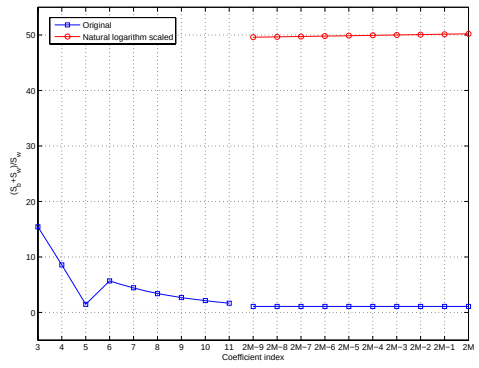


FIGURE 4.12: CRITERION FUNCTION VALUES FOR THE HOUSE DATASET.

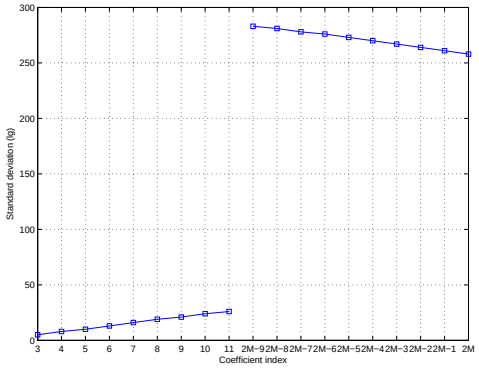


FIGURE 4.13: VARIANCE OF THE COEFFICIENTS FOR THE HOUSE DATASET.

We compute the Ihara coefficients for a hypergraph as described in Section 4.5 and construct a pattern vector of the form $\vec{v} = [c_3,\ c_4,\ \ln(|c_{M-3}|),\ \ln(|c_{M-2}|),\ \ln(|c_{M-1}|),\ \ln(|c_M|)]^T$. The reason for which we choose c_3 and c_4 as elements of the pattern vector is twofold. First, these two coefficients are more discriminative than the intermediate ones as

illustrated in Figure 4.12. Second and more importantly, c_3 and c_4 straightforwardly refer to some structural characteristics of the associated oriented line graph (i.e. c_3 is number of triangles and c_4 that of squares [49][79]). The last four components of the pattern vector are scaled in a logarithmic way to avoid problems of dynamic range. The other trailing coefficients are not taken as pattern vector elements due to their big values and great variance. For comparison, we use the truncated spectra of the Laplacian matrix introduced in Section 4.1.2 and the normalized Laplacian matrix described in [112]. The normalized Laplacian matrix of a hypergraph is defined as $\hat{\boldsymbol{L}}_H = \boldsymbol{I}_{|V|} - \boldsymbol{D}_v^{-1/2}\boldsymbol{H}\boldsymbol{D}_e\boldsymbol{H}^T\boldsymbol{D}_v^{-1/2}$, where $\boldsymbol{D}_v$ is the diagonal vertex degree matrix whose diagonal element $d(v_i)$ is the summation of the ith row of $\boldsymbol{H}$, and $\boldsymbol{D}_e$ is the diagonal vertex degree matrix whose diagonal element $d(e_j)$ is the summation of the jth column of $\boldsymbol{H}$. We take the second through to the seventh smallest eigenvalues of the Laplacian matrix, i.e. $\vec{v}_L = [\lambda_2^L,\ \lambda_3^L,\ \lambda_4^L,\ \lambda_5^L,\ \lambda_6^L,\ \lambda_7^L]^T$, and those of the normalized Laplacian matrix, i.e. $\vec{v}_{NL} = [\lambda_2^{NL},\ \lambda_3^{NL},\ \lambda_4^{NL},\ \lambda_5^{NL},\ \lambda_6^{NL},\ \lambda_7^{NL}]^T$, as the elements of pattern vectors. Figure 4.14 shows the Euclidean distance matrix for the three different pattern vectors, i.e. the truncated normalized Laplacian spectra, the truncated Laplacian spectra and the Ihara coefficients, on the House sequences. From Figure 4.14 it is clear that the truncated normalized Laplacian spectral vectors do not yield a strong block structure while the remaining two methods provide much better results. Figure 4.15 shows the PCA projections of the three different hypergraph pattern vectors for the Chalet house images. We explore whether the projection preserves the ordering of views which were obtained as the camera panned around the object. Each point in the pattern space is marked with a view number which corresponds to the order of the camera angle and the line connects consecutive object views. It is clear that the normalized Laplacian spectral method yields an erratic trajectory and the Laplacian spectral method provides a slightly clearer trajectory but is still hard to track. On the other hand, the Ihara coefficients produce a smoother trajectory and the neighboring images in the sequence are generally Euclidean neighbors in the eigenspace.

Figures 4.16(a), 4.16(b) and 4.16(c) illustrate the behavior of the leading non-zero eigenvalue of the normalized Laplacian matrix, the largest (final) eigenvalue of the Laplacian matrix and the final Ihara coefficient, respectively, of the first four objects in the COIL dataset. The four dotted lines represent the four objects separately. From Figure 4.16 it is clear that the second smallest eigenvalue of the normalized Laplacian matrix results in an overlap of all four objects. The largest eigenvalue of the Laplacian matrix results in an overlap for objects 1 and 2. On the other hand, when the final Ihara coefficient is used, the different objects are well separated. This reveals the potential of the Ihara coefficients as a means to effectively cluster objects.

We then embed the three different pattern vectors for hypergraphs extracted from the house sequences, the toy dataset and the COIL dataset into a three-dimensional space using PCA for the purposes of visualization. Figure 4.17 shows the results for the three classes of model houses. Figure 4.18 shows the results for the three classes of toys. Figures 4.19 shows the results for the first four objects in the COIL dataset. From Figures 4.17, 4.18 and 4.19 it is clear that the Ihara coefficients produce the best clusters, while the truncated normalized Laplacian spectra perform the worst of all among the three methods.

To take the quantitative evaluation of the pattern vectors one step further, we concentrate our attention on the COIL dataset, and evaluate the clustering performance obtained with different numbers of object classes. We compare the hypergraph Ihara coefficients with alternative hypergraph based methods (Laplacian spectra and normalized Laplacian spectra) as well as the method of spatial envelope [63]. Unlike the hypergraph based methods which use a high-level representation for visual information, the spatial envelope is a low-level pixel-based approach that represents the dominant spatial structure of a scene. For comparison, we first perform PCA on the pattern vectors established by using different methods. We then locate the clusters using the K-means method and calculate the Rand indices for the resulting clusters. The Rand indices for the three methods are listed in Table 4.1. From Table 4.1 it is clear that the Ihara coefficients outperform

Pattern Vector	Number of Object Classes			
	5	6	7	8
Spatial Envelope	0.7762	0.6775	0.7141	0.6944
Truncated Normalized Laplacian Spectra	0.7323	0.7074	0.7650	0.8030
Truncated Laplacian Spectra	0.8574	0.8564	0.8454	0.8449
Ihara Coefficients	0.9355	0.8859	0.8716	0.8812

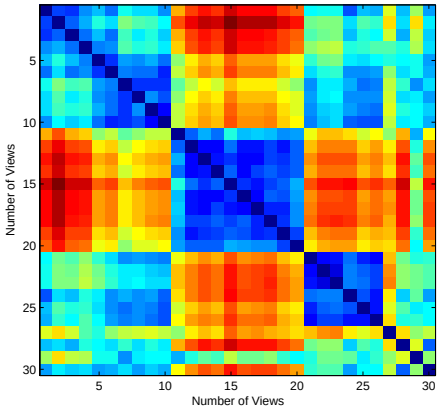
TABLE 4.1: RAND INDICES.

the alternative two spectral methods as well as spatial envelope for all numbers of object classes studied.

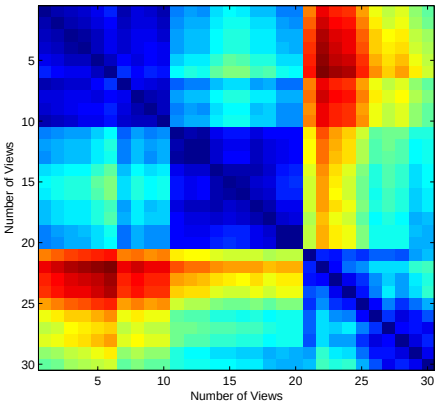
4.6.2 Computational Evaluation

In this subsection, we evaluate the computational efficiency of the algorithms for computing the hypergraph Ihara coefficients, and reveal the relationship between their computational overheads and the structural complexity of the associated graphs. The algorithms are tested on synthetic K-uniform hypergraphs, though the Ihara coefficients apply to nonuniform hypergraphs as well, because their structural complexity is easier to manipulate than randomly generated nonuniform hypergraphs. Each of the algorithms investigated were programmed using 32bit Matlab 2007a and run on a 2.40 GHz Intel(R) Core(TM)2 CPU with 3.24 GB of RAM.

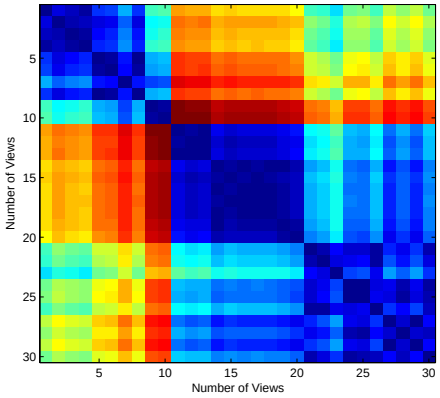
We commence by computing the hypergraph Ihara coefficients for a set hypergraphs with just one hyperedge and varying the number of vertices from 2 to 80. In other words we investigate the effect of varying the relational order. The time required to compute the coefficients based on the Perron-Frobenius operator $\boldsymbol{T}_H$ generated from the original hypergraph and the Perron-Frobenius operator $\boldsymbol{T}_{BG}$ generated from the associated bipartite graph are plotted separately as a function of relational order K (i.e. number of vertices



(a) Truncated normalized Laplacian spectra.

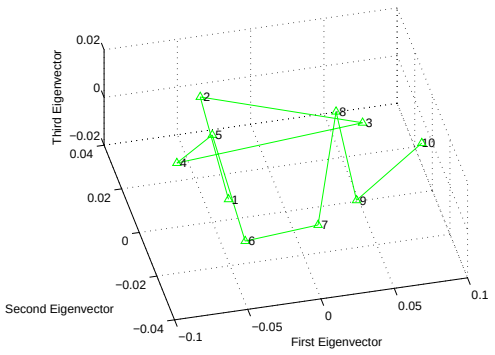


(b) Truncated Laplacian spectra.

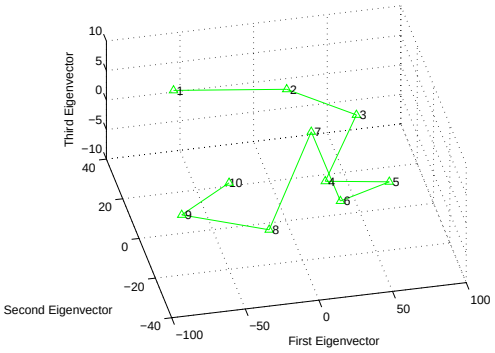


(c) Ihara coefficients.

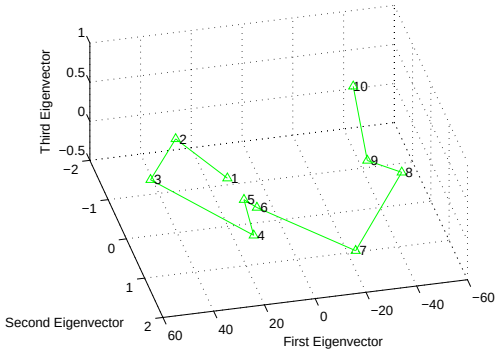
FIGURE 4.14: DISTANCE MATRIX.



(a) Truncated normalized Laplacian spectra.

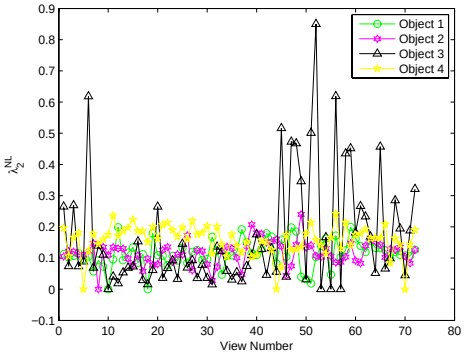


(b) Truncated Laplacian spectra.

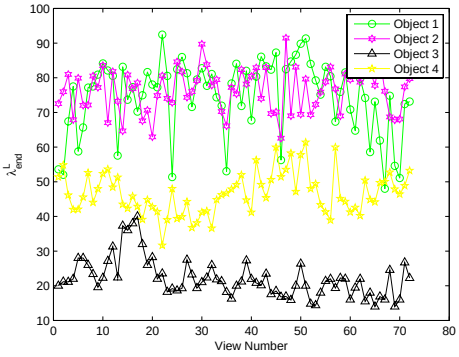


(c) Ihara coefficients.

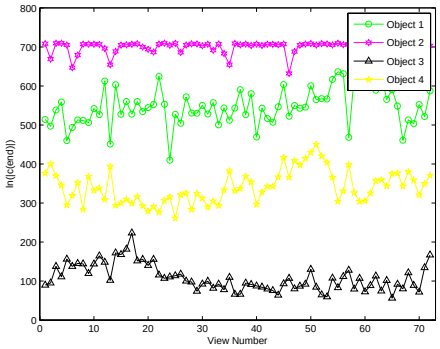
FIGURE 4.15: WITHIN-CLASS VIEW TRAJECTORY.



(a) Normalized Laplacian.

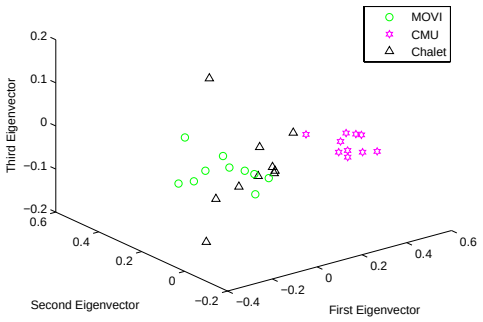


(b) Laplacian.

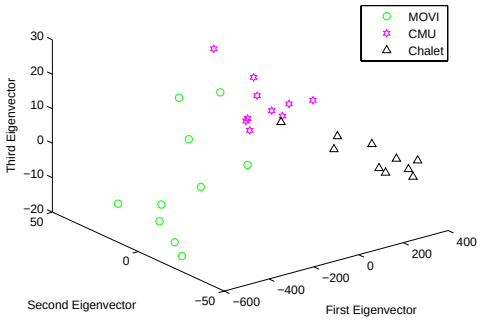


(c) Ihara coefficients.

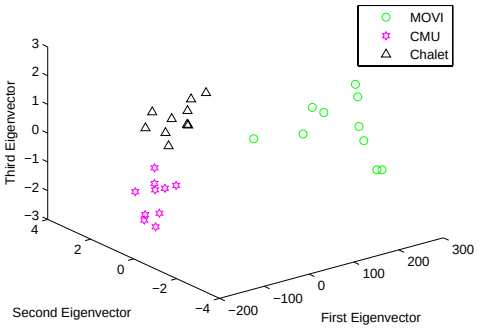
FIGURE 4.16: COEFFICIENT PLOT.



(a) Truncated normalized Laplacian spectra.

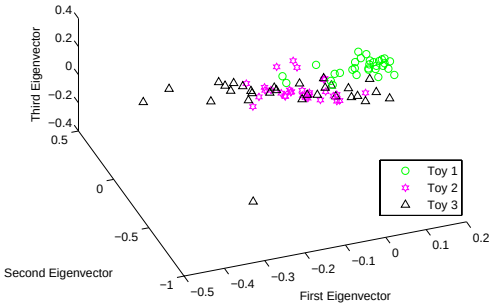


(b) Truncated Laplacian spectra.

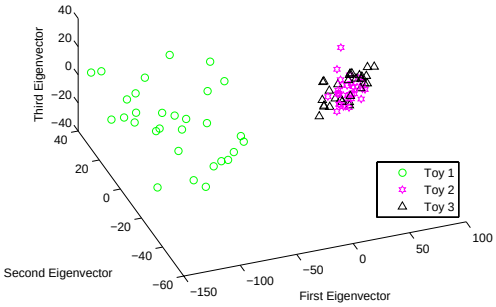


(c) Ihara coefficients.

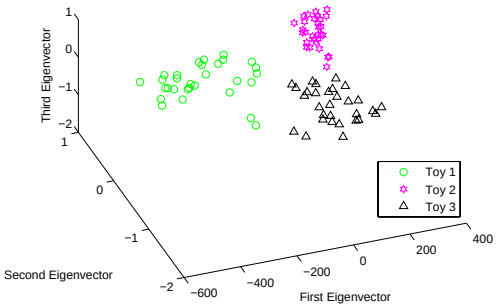
FIGURE 4.17: CLUSTERS FOR THREE CLASSES OF HOUSES.



(a) Truncated normalized Laplacian spectra.

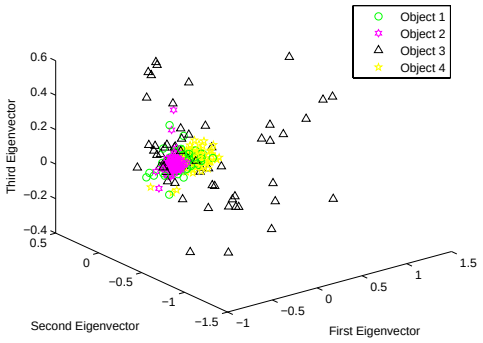


(b) Truncated Laplacian spectra.

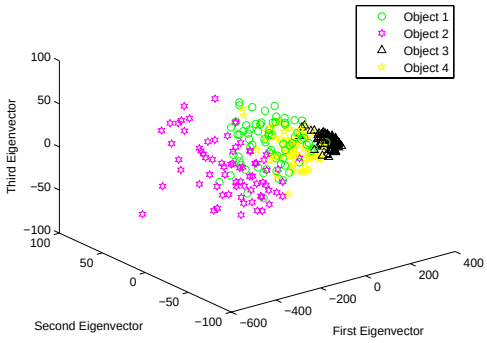


(c) Ihara coefficients.

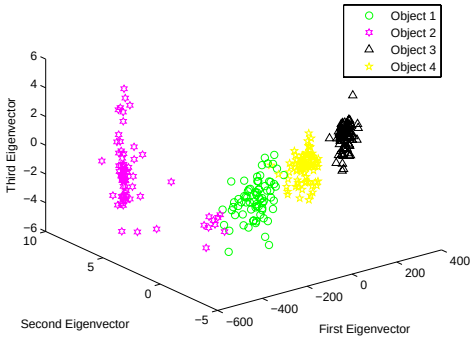
FIGURE 4.18: CLUSTERS FOR THREE CLASSES OF TOYS.



(a) Truncated normalized Laplacian spectra.



(b) Truncated Laplacian spectra.



(c) Ihara coefficients.

FIGURE 4.19: CLUSTERS FOR FOUR OBJECTS IN COIL DATASET.

participating in a hyperedge) in Figure 4.20. As the vertex cardinality of the hypergraph becomes larger from 2 up to 80, the time required to compute the coefficients based on $\boldsymbol{T}_H$ grows exponentially. However, the time required to compute the coefficients based on $\boldsymbol{T}_{BG}$ remains almost constant with increasing size of the relational order. Thus the computing time of the $\boldsymbol{T}_{BG}$-based method is less sensitive to relational order.

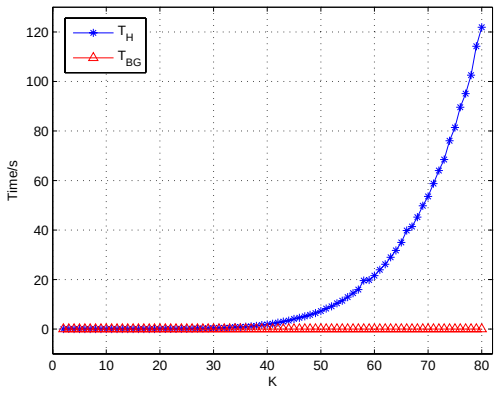


FIGURE 4.20: COMPUTING TIME FOR THE HYPERGRAPHS WITH ONE HYPEREDGE.

We then test the $\boldsymbol{T}_H$-based and $\boldsymbol{T}_{BG}$-based methods on K-uniform hypergraphs. We fix the vertex cardinality to 10 and establish K-uniform hypergraphs with the relational order K varying from 2 to 9. We confine the hypergraph established at each relational order to be complete, i.e. the hyperedges are the set of combinations of K different vertices. The total number of hyperedges, the total number of vertices of the oriented line graph generated from the original hypergraph, and the total number of vertices of the oriented line graph generated from the associated bipartite graph for each K-uniform hypergraph are shown in Figure 4.21. For the same K-uniform hypergraph, the vertex cardinality of the oriented line graph generated from the original hypergraph is greater than that from the associated bipartite graph.

The computing time for the methods based on the original hypergraph and the associated bipartite graph, i.e. $\boldsymbol{T}_H$ and $\boldsymbol{T}_{BG}$ respectively, are plotted separately as a function of

relational order K in Figure 4.22. For K-uniform hypergraphs, the trend for the computing time for the $\boldsymbol{T}_H$-based method is close to that of the vertex cardinality of the oriented line graph generated from the original hypergraph. This is also the case for computing time for $\boldsymbol{T}_{BG}$-based method and the vertex cardinality of the oriented line graph generated from the associated bipartite graph. Since the vertex cardinality of the former is greater than that of the latter, our proposed $\boldsymbol{T}_{BG}$-based method is certain to have lower computational overheads than the original method based on $\boldsymbol{T}_H$. This is supported by the results shown in Figure 4.22 and also accords with the complexity analysis presented in Section 4.5.

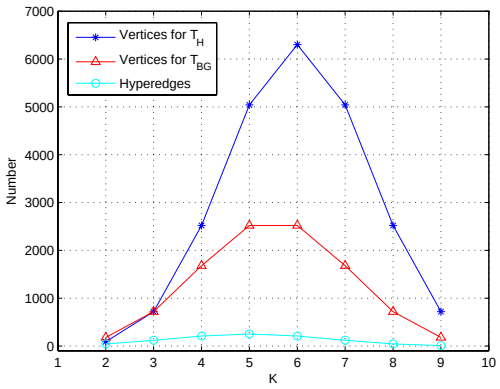


FIGURE 4.21: QUANTITIES IN THE REPRESENTATION.

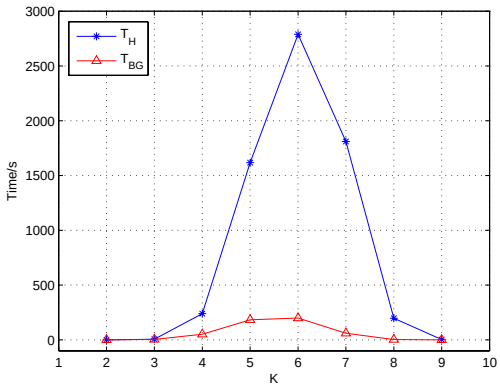


FIGURE 4.22: COMPUTING TIME FOR THE K-UNIFORM HYPERGRAPHS.

To evaluate the computational overheads of the algorithms for hypergraphs extracted from real-world images, we investigate the computing time of the $\boldsymbol{T}_H$- and $\boldsymbol{T}_{BG}$-based methods for the image sequence in Figure 4.11. Hypergraphs are constructed for the images as described in Section 4.6.1. The number of vertices and the average hyperedge cardinality for each hypergraph are presented in Table 4.2. Figure 4.23 illustrates the computing time for both methods. It is clear that the $\boldsymbol{T}_{BG}$-based method is significantly more efficient than the $\boldsymbol{T}_H$-based method for hypergraphs extracted from real-world images.

Image	(a)	(b)	(c)	(d)	(e)	(f)	(h)	(g)
Number of vertices	40	33	39	39	39	35	34	36
Average hyperedge cardinality	8.33	7.48	7.79	8.08	7.72	6.23	6.65	6.69

TABLE 4.2: SIZE OF HYPERGRAPHS EXTRACTED FROM IMAGES IN FIG .4.11.

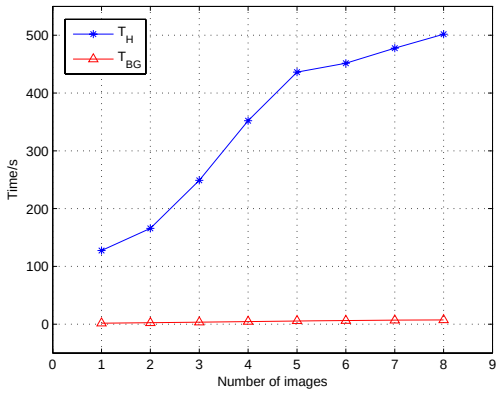


FIGURE 4.23: COMPUTING TIME FOR THE HYPERGRAPHS EXTRACTED FROM THE IMAGES OF THE TOY DUCKS.

4.7 Summary

In this chapter we have performed a characteristic polynomial analysis on hypergraphs and characterize (nonuniform) unweighted hypergraphs based on the Ihara zeta function. We have used the Ihara coefficients as the elements in the pattern vector for a hypergraph. We have also introduced an efficient method for computing the Ihara coefficients based on the associated bipartite graph. The Ihara coefficients are capable of distinguishing relational orders and thus can avoid the ambiguities caused by the hypergraph Laplacian. Experimental results show the effectiveness of the proposed method.

Chapter 5

High Order Structural Matching Using Dominant Cluster Analysis on a Direct Product Hypergraph

The idea described in this chapter is to extend the main cluster method of Leordeanu and Hebert [50] for graph matching and its generalization for high order matching [28], by using *dominant cluster analysis* (DCA) on a *direct product hypergraph* (DPH). For brevity we refer to the resulting algorithm as DPH-DCA. The novel aspect of our work resides in the use of the new concept of dominant cluster to develop an algorithm that not only outperforms the state-of-the-art methods but also satisfies the basic axioms of probability. We also present a method for initializing our algorithm that can be used to suppress outliers. This improves the matching performance of our method, and comparable results can not be achieved by using alternative high order matching algorithms [28][110]. Furthermore, we justify our DPH-DCA framework in terms of evolutionary game theory, and hereby observe that the optimal solution obtained by DPH-DCA achieves a Nash equilibrium subject to the Karush-Kuhn-Tucker (KKT) conditions. It is important to stress that we disregard the situation where feature labels are used as a reference for matching

(e.g. [14][108]) in the thesis. Instead, we concentrate our attention on the more challenging matching problem in which the only available information are the internal structural relationships between features within a set.

5.1 Problem Formulation

We consider the problem of matching two sets of features both with high order relationships. In each feature set, we refer to a subset of K elements as a K-tuple, and the Kth order relationships within the set are described in terms of measuring the similarity of each K-tuple. We thus represent the set of Kth order feature relationships by a K-uniform hypergraph $HG(V, E)$, whose hyperedges have identical cardinality K. Each vertex $v_i \in V$ in the K-uniform hypergraph $HG(V, E)$ represents one element in the feature set. Each hyperedge $e_i \in E$ represents one K-tuple $\{v_{i_1}, \cdots, v_{i_K}\} \in V$ and the weight attached to each hyperedge represents the similarity measure on the K-tuple encompassed by the hyperedge. For simplicity, we denote a vertex v_i by its index i in the remainder of our work. The K-uniform hypergraph $HG(V, E)$ can be represented as a Kth order tensor $\mathcal{H}$, whose element $H_{i_1,\cdots,i_K}$ is the hyperedge weight if there is a hyperedge encompassing the vertex subset $\{i_1, \cdots, i_K\} \in V$, and zero otherwise. The problem of matching two feature sets both constituted by Kth order relationships can then be transformed to that of matching the two associated K-uniform hypergraphs $HG(V, E)$ and $HG'(V', E')$. To this end, we establish a high order compatibility tensor $\mathcal{C}$, i.e. compatibility tensor, for $HG(V, E)$ and $HG'(V', E')$. The elements of the Kth order compatibility tensor $\mathcal{C}$ are defined as follows

$$C_{i_1 i'_1,\cdots,i_K i'_K} = \begin{cases} 0 & \text{if } H_{i_1,\cdots,i_K} = 0 \text{ or } H'_{i'_1,\cdots,i'_K} = 0; \\ s(H_{i_1,\cdots,i_K}, H'_{i'_1,\cdots,i'_K}) & \text{otherwise;} \end{cases} \tag{5.1}$$

where $s(\cdot,\cdot)$ is a function that measures the similarity between a pair of hyperedges, and there are many alternative methods that are available in literature to define the function. In our study, we adopt the definition $s(H_{i_1,\cdots,i_K}, H'_{i'_1,\cdots,i'_K}) = \exp(-\|H_{i_1,\cdots,i_K} - H'_{i'_1,\cdots,i'_K}\|_2^2/\sigma_1)$ where σ_1 is a scaling parameter. This function derives from the Gaussian kernel for measuring similarity between features and has been widely used in the matching scenario [25][28][50]. Each element of the compatibility tensor $\mathcal{C}$ represents a similarity measure between the two corresponding hyperedges. The hyperedge pair $\{i_1,\cdots,i_K\}$ and $\{i'_1,\cdots,i'_K\}$ with a large similarity measure has a large probability $\Pr(\{i_1,\cdots,i_K\} \leftrightarrow \{i'_1,\cdots,i'_K\}|H,H')$ for matching. Here the notation $\leftrightarrow$ denotes a possible matching between a pair of hyperedges or a pair of vertices. Under the conditional independence assumption of the matching process [110], the probability for hyperedge correspondence can be factorized over the associated vertices of the hyperedges as follows

$$\Pr(\{i_1,\cdots,i_K\} \leftrightarrow \{i'_1,\cdots,i'_K\}|HG,HG') = \prod_{n=1}^{K} \Pr(i_n \leftrightarrow i'_n|HG,HG') \qquad (5.2)$$

where $\Pr(i_n \leftrightarrow i'_n|HG,HG')$ denotes the probability for the possible matching $i_n \leftrightarrow i'_n$ to be correct. For two hypergraphs $HG(V,E)$ and $HG(V',E')$ with $|V| = N$ and $|V'| = N'$ respectively, we denote their $N \times N'$ matching matrix by $\mathbf{P}$ with the (i,i')th entry $P_{ii'} = \Pr(i \leftrightarrow i'|HG,HG')$. According to (5.2), high order matching problems can be formulated as locating the matching probability that most closely accords with the elements of the compatibility tensor, i.e. seeking the optimal $\mathbf{P}$ by maximizing the following objective function

$$f(\mathbf{P}) = \sum_{i_1=1}^{N}\sum_{i'_1=1}^{N'}\cdots\sum_{i_K=1}^{N}\sum_{i'_K=1}^{N'} C_{i_1 i'_1,\cdots,i_K i'_K}\Pr(\{i_1,\cdots,i_K\}\leftrightarrow\{i'_1,\cdots,i'_K\}|HG,HG')$$
$$= \sum_{i_1=1}^{N}\sum_{i'_1=1}^{N'}\cdots\sum_{i_K=1}^{N}\sum_{i'_K=1}^{N'} C_{i_1 i'_1,\cdots,i_K i'_K}\prod_{n=1}^{K} P_{i_n i'_n} \tag{5.3}$$

subject to $\forall i,j,\ P_{ii}\geq 0$ and $\sum_{i=1}^{N}\sum_{i'=1}^{N'} P_{ii'}=1$. The objective function (5.3) is the correlation between the hyperedge similarity $C_{i_1 i'_1,\cdots,i_K i'_K}$ and the K-wise product $\prod_{n=1}^{K} P_{i_n i'_n}$ for vertex correspondences. It is a natural extension of the objective function that is commonly used in classical pairwise matching algorithms such as softassign [35]. It generalizes the relational order from two to an arbitrary high order K. This formulation has also been adopted in tensor power iteration for higher order matching [28]. However, the difference between our method and the existing algorithms is that we restrict the solution of (5.3) to obey the the fundamental axioms of probability, i.e. positiveness and unit total probability mass. This constraint not only provides an alternative probabilistic perspective for hypergraph matching, but also proves convenient for optimization. We denote the optimal solution matrix of (5.3) by $\widehat{\mathbf{P}}$ with the (i,i')th entry $\widehat{P}_{ii'}=\widehat{\Pr}(i\leftrightarrow i'|HG,HG')$. We refer to $\widehat{\Pr}(i\leftrightarrow i'|HG,HG')$ as the matching probability for vertex i and i'. The set of matching probabilities $\{\widehat{\Pr}(i\leftrightarrow i'|HG,HG')|i\in V; i'\in V'\}$ maximizing (5.3) indicate how likely it is that each correspondence is correct according to structural similarity between the two hypergraphs HG and HG'.

On the other hand, we also observe that the objective function (5.3) can be justified in terms of evolutionary game theory. In this regard we will further describe how the process of optimizing (5.3) achieves a Nash equilibrium subject to the Karush-Kuhn-Tucker (KKT) conditions. This observation provides a different perspective for understanding the principles underlying our high order matching framework. The details regarding the evolutionary game theoretic interpretation will be presented in Section 5.4.

Once the set of matching probabilities satisfying (5.3) are computed, correspondences

between vertices drawn from HG and HG' can be established. Matchings with a zero probability are the least likely correspondences, and matchings with nonzero probabilities tend to be those with significant similarity between their structural contexts. Our aim is to seek the subset of possible correspondences with non-zero matching probabilities which satisfy (5.3), that are subject to the one-to-one matching constraint.

5.2 High Order Matching As Dominant Cluster Analysis on a Direct Product Hypergraph

In this section we pose the high order relational matching problem formulated in (5.3) as one of *dominant cluster analysis* (DCA) on a *direct product hypergraph* (DPH). We commence by establishing a direct product hypergraph for the two hypergraphs, which separately represent the two sets of high order features to be matched. Optimal matching can be achieved by extracting the dominant cluster of vertices from the direct product hypergraph.

5.2.1 Direct Product Hypergraph

The construction of a direct product hypergraph for two K-uniform hypergraphs is a generalization of that of the direct product graph [97], which can be used to construct kernels for graph classification. We extend the concept of a direct product graph to encapsulate high order relations residing in a hypergraph and apply this generalization to hypergraph matching problems. For two K-uniform hypergraphs $HG(V, E)$ and $HG'(V', E')$, the direct product $HG_\times$ is a hypergraph with vertex set

$$V_\times = \{(i, i') | i \in V, i' \in V'\}; \tag{5.4}$$

and edge set

$$E_\times = \{\{(i_1, i'_1) \cdots (i_K, i'_K)\} | \{i_1, \cdots, i_K\} \in E, \{i'_1, \cdots, i'_K\} \in E'\}. \tag{5.5}$$

The vertex set of the direct product hypergraph $HG_\times$ consists of Cartesian pairs of vertices drawn from HG and HG' separately. Thus the cardinality of the vertex set of $HG_\times$ is $|V_\times| = |V||V'| = NN'$. The direct product hypergraph $HG_\times$ is K-uniform, and each K-tuple of vertices in $HG_\times$ is encompassed in a hyperedge if and only if the corresponding vertices in HG and HG' are both encompassed by a hyperedge in the relevant hypergraph. Each hyperedge in a direct product hypergraph is weighted by the similarity between the two associated hyperedges from HG and HG'. Two example hypergraphs and their direct product hypergraph are shown in Figure 5.8 and Table 5.2 respectively.

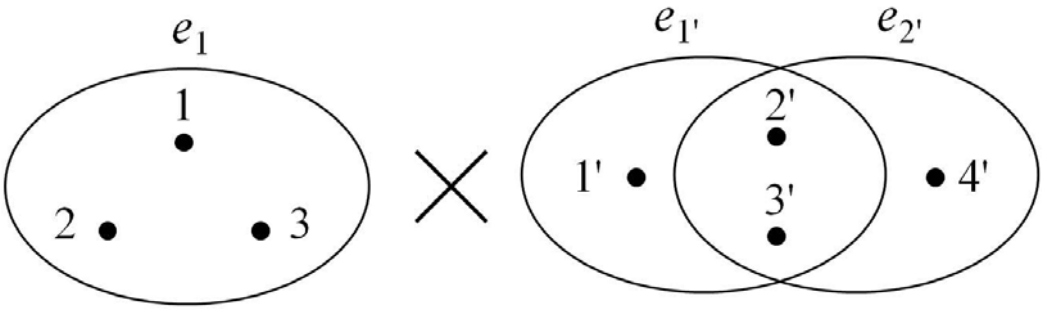


FIGURE 5.8: TWO EXAMPLE HYPERGRAPHS (LEFT AND RIGHT).

Hyperedge index	1	2	3	4	5	6	7	8	9	10	11	12
Vertex indices	$11'$	$11'$	$12'$	$12'$	$13'$	$13'$	$12'$	$12'$	$13'$	$13'$	$14'$	$14'$
	$22'$	$23'$	$21'$	$23'$	$21'$	$22'$	$23'$	$24'$	$22'$	$24'$	$22'$	$23'$
	$33'$	$32'$	$33'$	$31'$	$32'$	$31'$	$34'$	$33'$	$34'$	$32'$	$33'$	$32'$
Hyperedge weight	$s(e_1, e_{1'})$						$s(e_1, e_{2'})$					

TABLE 5.2: DIRECT PRODUCT HYPERGRAPH OF THE TWO HYPERGRAPHS IN FIGURE 5.8. (FOR SIMPLICITY WE INDEX EACH VERTEX $(i, i') \in V_\times$ BY ii'.)

In Table 5.2 we can see that there is no hyperedge encompassing $\{(1, 1'), (2, 2'), (3, 4')\}$

in the direct product hypergraph. This is because $\{1', 2', 4'\}$ is not encompassed by a hyperedge in the right example hypergraph in Figure 5.8 .

Furthermore, from our definition of direct product hypergraph, it is clear that the compatibility tensor $\mathcal{C}$ defined in (5.1) is in fact the tensor $\mathcal{C}_\times$ associated with the direct product hypergraph $HG_\times$ for HG and HG'. Every possible matching $i \leftrightarrow i'$ is associated with the vertex (i, i') in $HG_\times$. For simplicity we let α denote the vertex represented by the Cartesian pair (i, i'), and let $\mathbb{D}$ denote the subset of vertices in $HG_\times$ which represent the correct vertex matching for HG and HG'. We denote the probability for the vertex α belonging to $\mathbb{D}$ by $\Pr(\alpha \in \mathbb{D}|HG_\times)$. For a direct product hypergraph with $N_\times$ vertices, we establish an $N_\times \times 1$ vector $\mathbf{p}$ with the αth element $p_\alpha = \Pr(\alpha \in \mathbb{D}|HG_\times)$. With these ingredients the optimal model satisfying the condition (5.3) reduces to

$$\widehat{\mathbf{p}} = \underset{\mathbf{p}}{\operatorname{argmax}} \sum_{\alpha_1=1}^{N_\times} \cdots \sum_{\alpha_K=1}^{N_\times} C_{\alpha_1,\cdots,\alpha_K} \prod_{n=1}^{K} p_{\alpha_n} \tag{5.6}$$

subject to the constraints $\forall \alpha,\ p_\alpha \geq 0$ and $\sum_{\alpha=1}^{N_\times} p_\alpha = 1$. According to (5.6), zero probability will be assigned to the vertices that do not belong to $\mathbb{D}$. We refer to the probability $\widehat{\Pr}(\alpha \in \mathbb{D}|HG_\times) = \widehat{p}_\alpha$ where $\widehat{p}_\alpha$ is the αth element of the vector $\widehat{\mathbf{p}}$ satisfying the optimality condition in (5.6) as the association probability for the vertex α. Therefore, the matching problem can be solved by extracting the cluster of vertices with nonzero association probabilities in the direct product hypergraph.

5.2.2 Dominant Cluster Analysis

In this subsection, we present a route to obtaining the optimal matching by clustering the vertices in the direct product hypergraph. Drawing on the concept of the dominant set in a graph [64] and its game theoretic generalization [73], we commence by defining the average similarity of a subset of vertices in the direct product hypergraph.

Definition 1. *The average similarity of a subset $\widetilde{V}_\times$ of vertices in a K-uniform direct product hypergraph $HG_\times(V_\times, E_\times)$ is defined as*

$$S(\widetilde{V}_\times) = \sum_{\{\alpha_1,\cdots,\alpha_K\}\subset\widetilde{V}_\times} C_{\alpha_1,\cdots,\alpha_K} \prod_{n=1}^{K} w_{\alpha_n} \tag{5.7}$$

subject to the positiveness constraint $w_\alpha \geq 0$ for $\alpha \in V_\times$ and the normalization constraint $\sum_{\alpha\in V_\times} w_\alpha = 1$, where $C_{\alpha_1,\cdots,\alpha_K}$ is the element of the tensor $\mathcal{C}_\times$ for $HG_\times(V_\times, E_\times)$, the set $\{\alpha_1, \cdots, \alpha_K\}$ represents a K-tuple in $V_\times$, and w_α is the association weight assigned to the vertex α.

Theorem 1. *For a K-regular direct product hypergraph $HG_\times(V_\times, E_\times)$, if we set the association weight of a vertex equal to the association probability, i.e. $w_\alpha = \widehat{\Pr}(\alpha \in \mathbb{D}|HG_\times)$ which maximizes the objective function (5.6), the subset of vertices with non-zero weights have the largest average similarity over all possible subsets of vertices and all possible vertex weighting assignments.*

Proof. Let $\widehat{V}_\times \subseteq V_\times$ denote the vertex subset that is associated with non-zero association probabilities, then we have the following properties for $\alpha \in V_\times$ and the subset $\widehat{V}_\times$

i) $\forall\alpha \in \widehat{V}_\times, w_\alpha = \widehat{\Pr}(\alpha \in \mathbb{D}|HG_\times) > 0$;

ii) $\forall\alpha \notin \widehat{V}_\times$ and $\alpha \in V_\times,\ w_\alpha = \widehat{\Pr}(\alpha \in \mathbb{D}|HG_\times) = 0$.

As a result of these two properties, the average similarity of $\widehat{V}_\times$ is

$$\begin{aligned} S(\widehat{V}_\times) &= \sum_{\{\alpha_1,\cdots,\alpha_K\}\subset\widehat{V}_\times} C_{\alpha_1,\cdots,\alpha_K} \prod_{n=1}^{K} w_{\alpha_n} \\ &= \sum_{\alpha_1=1}^{N_\times} \cdots \sum_{\alpha_K=1}^{N_\times} C_{\alpha_1,\cdots,\alpha_K} \prod_{n=1}^{K} \widehat{\Pr}(\alpha_n \in \mathbb{D}|HG_\times) \end{aligned} \tag{5.8}$$

Equation (5.8) holds because for any K-tuple including one vertex $\alpha_n \notin \widehat{V}_\times$ the term $C_{\alpha_1,\cdots,\alpha_K} \prod_{n=1}^{K} \widehat{\Pr}(\alpha_n \in \mathbb{D}|HG_\times)$ is equal to zero.

It is clear that the average similarity of $\widehat{V}_\times$ is associated with the optimal solution of (7), which runs over all possible weighting assignments. As a result, the subset of vertices

with non-zero association probabilities as weights have the largest average similarity over all possible subsets of vertices and all possible vertex weighting assignments. □

Definition 2. *A dominant cluster for a direct product hypergraph is the subset of vertices whose association probabilities are not zero.*

Based on Definitions 1 and 2 and Theorem 1, it is clear that the problem of high order matching can be straightforwardly transformed to a process of extracting the dominant cluster from the direct product hypergraph. We refer to this process as *dominant cluster analysis* on a *direct product graph* (DPH-DCA).

Let **w** denote a vector with the αth element w_α. According to the iterative technique for maximization reported in [5], the maximization of a homogeneous polynomial $g(\mathbf{w})$ with the constraints that $\forall \alpha, w_\alpha > 0$ and $\sum_\alpha w_\alpha = 1$ can be achieved by applying the following update until converged

$$w_\alpha^{\mathtt{new}} = w_\alpha \frac{\partial g(\mathbf{w})}{\partial w_\alpha} / \sum_{\beta=1}^{N_\times} w(\beta) \frac{\partial g(\mathbf{w})}{\partial w_\beta} \qquad \forall \alpha. \tag{5.9}$$

In the matching scenario, we define $g(\mathbf{w}) = \sum_{\alpha_1=1}^{N_\times} \cdots \sum_{\alpha_K=1}^{N_\times} C_{\alpha_1,\cdots,\alpha_K} \prod_{n=1}^{K} w_{\alpha_n}$. According to the definition of the compatibility tenor in (5.1), $C_{\alpha_1,\cdots,\alpha_K} = 0$ if in $\alpha_1, \cdots, \alpha_K$ there is a pair of indices equal to each other. As a result, the partial derivative of $g(\mathbf{w})$ can be computed as follows

$$\frac{\partial g(\mathbf{w})}{\partial w_\alpha} = K \sum_{\alpha_2=1}^{N_\times} \cdots \sum_{\alpha_K=1}^{N_\times} C_{\alpha,\alpha_2,\cdots,\alpha_K} \prod_{n=2}^{K} w_{\alpha_n} \tag{5.10}$$

Substituting (5.10) into (5.9), we can easily perform DPH-DCA by applying the following update until convergence is reached

$$w_\alpha^{\mathtt{new}} = \frac{w_\alpha \sum_{\alpha_2=1}^{N_\times} \cdots \sum_{\alpha_K=1}^{N_\times} C_{\alpha,\alpha_2,\cdots,\alpha_K} \prod_{n=2}^{K} w_{\alpha_n}}{\sum_{\beta=1}^{N_\times} w_\beta \sum_{\beta_2=1}^{N_\times} \cdots \sum_{\beta_K=1}^{N_\times} C_{\beta,\beta_2,\cdots,\beta_K} \prod_{n=2}^{K} w_{\beta_n}} \tag{5.11}$$

At convergence we obtain the optimal weight vector $\widehat{\mathbf{w}} = [\widehat{w}_1, \widehat{w}_2, \cdots, \widehat{w}_{N_\times}]^T$, with the αth component $\widehat{w}_\alpha$ representing the the optimal association weight for the vertex α

belonging to the dominant cluster $\mathbb{D}$. Since α is the simplified donation for the Cartesian pair (i, i'), $\widehat{w}_\alpha$ also represents the optimal score for $i \leftrightarrow i'$ to be a correct matching. In other words, $\widehat{\mathbf{p}} = \widehat{\mathbf{w}}$ is an optimal solution for the objective function (5.6) and its original form (5.3).

5.3 Matching with Prior Rejections

The high order structural matching algorithm described in Section 5.2 is an unsupervised process. The weight of each vertex in the direct product hypergraph can be initialized by using a uniform distribution of probability. However, if two vertices in a hypergraph have the same structural context, i.e. their interchange does not change the hypergraph structure, they can cause ambiguity when matching is attempted. Two alternative state-of-the-art methods, namely probabilistic hypergraph matching [110] and tensor power iteration [28], also suffer from this shortcoming. An example of the ambiguity is shown in Figure 5.9, in which the vertices 2 and 5 have the same structural context within the left hypergraph. We compute the compatibility tensor $\mathcal{C}$ for the two hypergraphs in Figure 5.9, according to (5.1) with $\sigma_1 = 1$. If we adopt the initialization of weighting each vertex by a uniform probability $1/20$, the matching scores computed by using different methods for all the 20 possible matchings are shown in Table 5.3 (disregarding the row (d) for the time being). It is clear that although the three methods adopt different optimization strategies, each of them results in the same matching score for $2 \leftrightarrow 2'$ and $5 \leftrightarrow 2'$, and can not distinguish which one of the two possible matchings is more likely to be correct. This is misleading because only one of the two possibilities is allowed to be correct under the one-to-one matching constraint. The reason for this ambiguity within the DPH-DCA framework is that the matching scores are initialized by a uniform distribution of probability for the update formula (5.11) and this results in a matching ambiguity for vertices with the same structural context in a hypergraph.

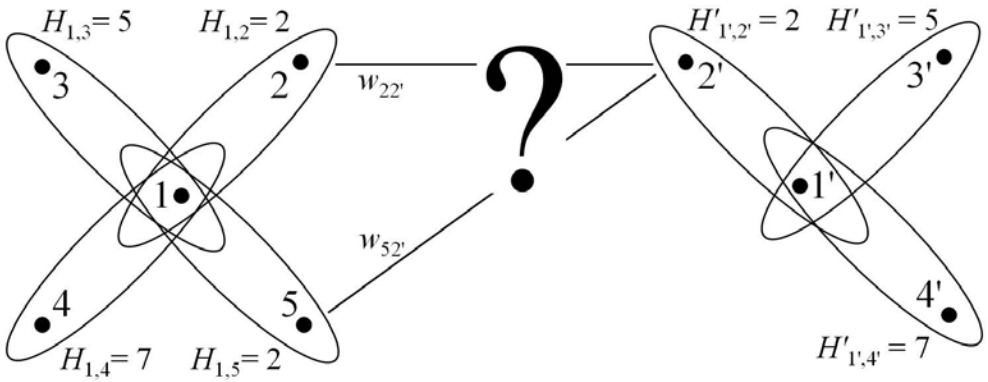


FIGURE 5.9: EXAMPLE HYPERGRAPHS CAUSING MATCHING AMBIGUITIES.

However, if prior knowledge about outliers (i.e. hypergraph vertices for which no match exists) is available, we can to a certain extent avoid the ambiguity and improve matching accuracy by using a different initialization strategy. We refer to the vertex subset $V_\times^o \subseteq V_\times$ (i.e. possible correspondences) associated with available outliers as prior rejections, and the adopted initialization in the light of prior rejections is as follows

$$w(\alpha) = \begin{cases} 0 & \text{if } \alpha \in V_\times^o; \\ 1/(N_\times - N_\times^o) & \text{otherwise;} \end{cases} \tag{5.12}$$

where $N_\times^o$ is the cardinality of $V_\times^o$. The matching scores for the hypergraph examples in Figure 5.9 are shown in the row (d) of Table 5.3, where we assume that the vertex 5 is an identified outlier. It is clear that $5 \leftrightarrow 2'$ maintains a zero score and does not necessarily influence the matching score for $2 \leftrightarrow 2'$ any more.

The initialization scheme (5.12) improves the matching accuracy within the DPH-DCA framework because the vertex weight w_α in the numerator of the update formula (5.11) plays an important role in maintaining the initial rejection. It enables the prior rejections to maintain a zero weight and does not affect the matching scores for other possible correspondences at each update until converged. The extent to which the matching accuracy can be improved depends on the amount of prior rejections available. The more

	$1 \leftrightarrow 1'$	$2 \leftrightarrow 1'$	$3 \leftrightarrow 1'$	$4 \leftrightarrow 1'$	$5 \leftrightarrow 1'$	$1 \leftrightarrow 2'$	$2 \leftrightarrow 2'$	$3 \leftrightarrow 2'$	$4 \leftrightarrow 2'$	$5 \leftrightarrow 2'$
(a)	0.1866	0.1866	0.1866	0.1866	0.1866	0.0938	0.2401	0.2208	0.2208	0.2401
(b)	0.1971	0	0	0	0	0	0.1989	0	0	0.1989
(c)	0.5000	0	0	0	0	0	0.1250	0	0	0.1250
(d)	0.5000	0	0	0	0	0	0.1667	0	0	0
	$1 \leftrightarrow 3'$	$2 \leftrightarrow 3'$	$3 \leftrightarrow 3'$	$4 \leftrightarrow 3'$	$5 \leftrightarrow 3'$	$1 \leftrightarrow 4'$	$2 \leftrightarrow 4'$	$3 \leftrightarrow 4'$	$4 \leftrightarrow 4'$	$5 \leftrightarrow 4'$
(a)	0.0782	0.2283	0.2462	0.2276	0.2283	0.0782	0.2283	0.2276	0.2462	0.2283
(b)	0	0	0.1989	0.0036	0	0	0	0.0036	0.1989	0
(c)	0	0	0.1250	0	0	0	0	0	0.1250	0
(d)	0	0	0.1667	0	0	0	0	0	0.1667	0

TABLE 5.3: MATCHING SCORES FOR THE TWO EXAMPLE HYPERGRAPHS IN FIGURE 5.9 COMPUTED BY USING THE ALTERNATIVE METHODS (A) PROBABILISTIC HYPERGRAPH MATCHING, (B) TENSOR POWER ITERATION, (C) DPH-DCA WITHOUT PRIOR REJECTIONS AND (D) DPH-DCA WITH PRIOR REJECTIONS.

prior knowledge concerning the outliers that is available, the more accurate the matching that can be obtained. This will be verified in our experimental section.

However, the initialization scheme (5.12) does not apply to the two alternative methods described in the literature [110][28]. The probabilistic hypergraph matching method [110] initializes a matching score by a fixed value obtained from the marginalization of the compatibility tensor, and thus can not accommodate the prior rejections by using (5.12). The tensor power iteration method [28], though manually initialized, converges to a fixed matching score for different initializations. As a result, the two alternative methods are unable to accommodate prior rejections in the same manner as our DPH-DCA framework.

5.4 An Evolutionary Game Theoretic Interpretation

In this section, we provide an evolutionary game theoretic perspective for the DPH-DCA framework. We are motivated by the recent work of game theoretic approach to visual correspondence [3], in which the problem of graph matching is posed as that of a non-cooperative game. The main difference is that the DPH-DCA framework can be applied to establishing both pairwise and higher order correspondences while the game theoretic approach presented in [3] is restricted to pairwise graphs. The game-theoretic analysis has also been incorporated into a clustering scenario [73]. On the other hand, we adopt evolutionary game theory here as a justification of our high order matching framework rather than that of clustering.

The general evolutionary game theory has been comprehensively investigated in a vast variety of research fields such as sociology, economics and biology. In an evolutionary game, a population of independent players simultaneously select their preferred strategies, and each player receives a payoff proportional to the compatibility of the selected strategy with respect to the strategies selected by the other players. The evolution within a game is a selection mechanism that spreads the fittest strategies in the population, and accordingly drives the detrimental ones to extinction. The evolution will finally result in a state of evolutionarily stable strategies that maximizes the average payoff over the population.

We justify the DPH-DCA framework for structural matching in terms of strategic interactions (i.e. evolutionary games) among a population of players. Given two sets of Kth order relational features, we refer to each possible correspondence between two features which are drawn from the two sets separately as a strategy. Based on the formulation of the DPH-DCA framework presented in Section 5.2, we assume there is a game with $N_\times$ strategies. We denote the probability distribution over all strategies by an $N_\times$-dimensional vector $\mathbf{p}$ with the αth component p_α representing the probability for an independent player to select the αth strategy, subject to the conditions a) $\forall \alpha,\ p_\alpha \geq 0$ and b) $\sum_{\alpha=1}^{N_\times} p_\alpha = 1$. Additionally, the element $C_{\alpha_1,\cdots,\alpha_K}$ of the compatibility tensor $\mathcal{C}_\times$ is used to provide a

payoff for players selecting the strategies $\alpha_1, \cdots, \alpha_K$. The overall average payoff with respect to K independent players involved in the game is

$$g(\mathbf{p}) = \sum_{\alpha_1=1}^{N_\times} \cdots \sum_{\alpha_K=1}^{N_\times} C_{\alpha_1,\cdots,\alpha_K} \prod_{n=1}^{K} p_{\alpha_n}. \tag{5.13}$$

The overall average payoff (5.13) can also be formulated in terms of the tensor notation. To this end, we define the tensor product function with respect to the compatibility tensor $\mathcal{C}_\times$ and K probability vectors $\mathbf{p}_1, \cdots, \mathbf{p}_K$ as follows

$$G(\mathbf{p}_1, \cdots, \mathbf{p}_K) = \mathcal{C}_\times \times_1 \mathbf{p}_1^T \times_2 \mathbf{p}_2^T \cdots \times_K \mathbf{p}_K^T. \tag{5.14}$$

Any interchanges of the vectors in (5.14) do not affect the value of the tensor product due to the hypersymmetric property of the compatibility tensor $\mathcal{C}_\times$. See [96] for more details about the product for a tensor.

Let $\mathbf{p}_i = \mathbf{p}, \forall\, \mathbf{p}_i \in \{\mathbf{p}_1, \cdots, \mathbf{p}_K\}$, and we have the the overall average payoff (5.13) reformulated in terms of the tensor notation as follows

$$g(\mathbf{p}) = G(\overbrace{\mathbf{p}, \cdots, \mathbf{p}}^{K \text{ players}}) = \mathcal{C}_\times \times_1 \mathbf{p}^T \times_2 \mathbf{p}^T \cdots \times_K \mathbf{p}^T. \tag{5.15}$$

In the evolution within a game, the probability vector $\mathbf{p}$ is expected to achieve the state of evolutionarily stable strategies thereby in the most favor of maximizing the overall average payoff (5.15). On the other hand, it is worth noting that the relations in (5.13) and (5.15) are in fact identical to that in the objective function (5.3) for the DPH-DCA framework. Additionally, the update (5.11) behaves in a sense as the process of evolution, in which the strategies favoring the overall average payoff tend to be selected and thus survive every round of evolution while the unfavorable strategies become less preferable and finally extinct in the evolution. This can be verified by the fact that the update (5.11) at convergence only maintains the most favorable weights contributing to the maximization of (5.6) with non-zero values and drives the remaining less favorable weights to zero. One concrete example can be seen in the rows (c) and (d) of Table 5.3, where only

weights associated with optimal matchings exhibit non-zero values. Correspondences associated with the identified outliers (prior rejections) can be regarded as strategies which are acknowledged by all independent players as unfavorable choices, and they are thus not supposed to be selected by any independent player. Therefore, according to the discussions in Section 5.2, the optimal matching probability vector $\widehat{\mathbf{p}}$ that maximizes the objective function (5.3) also provides the evolutionarily stable strategies for the evolutionary game with respect to the compatibility $\mathcal{C}_\times$. This observation not only justifies our objective function (5.3) in terms of evolutionary game theory but also provides a novel perspective for high order structural matching.

For a further exploration of evolutionary game theory in the high order structural matching scenario, we investigate the Lagrange function for maximizing overall average payoff (5.13). Let $u_1, \cdots, u_{N_\times}$ and λ be Lagrange multipliers and $\mathbf{u} = [u_1, \cdots, u_{N_\times}]^T$, and the Lagrange function with respect to $\mathbf{p}$ is

$$L(\mathbf{p}, \mathbf{u}, \lambda) = G(\overbrace{\mathbf{p}, \cdots, \mathbf{p}}^{K \text{ players}}) + \mathbf{u} \cdot \mathbf{p} + \lambda(1 - \sum_{\alpha=1}^{N_\times} p_\alpha). \tag{5.16}$$

The optimal probability vector $\widehat{\mathbf{p}}$ maximizing the overall average payoff (5.15) is supposed to satisfy the relation

$$\frac{\partial G(\overbrace{\mathbf{p}, \cdots, \mathbf{p}}^{K \text{ players}})}{\partial p_\alpha}\Big|_{\mathbf{p}=\widehat{\mathbf{p}}} + u_\alpha \widehat{p}_\alpha - \lambda = 0, \tag{5.17}$$

for $1 \leq \alpha \leq N_\times$. Let $\mathbf{e}_\alpha$ denote an $N_\times$-dimensional vector with the αth element 1 and the remaining elements 0. According to the partial derivative (5.18) of the objective function

(5.6), we have the partial derivative of the overall average payoff (5.15) as follows

$$\begin{aligned}\frac{\partial G(\overbrace{\mathbf{p},\cdots,\mathbf{p}}^{K \text{ players}})}{\partial p_\alpha}\Big|_{\mathbf{p}=\widehat{\mathbf{p}}} =& K\sum_{\alpha_2=1}^{N_\times}\cdots\sum_{\alpha_K=1}^{N_\times} C_{\alpha,\alpha_2,\cdots,\alpha_K}\prod_{n=2}^{K}\widehat{p}_{\alpha_n}\\ =& \mathcal{C}_\times\times_1\widehat{\mathbf{p}}^T\times_2\widehat{\mathbf{p}}^T\cdots\times_\alpha\mathbf{e}_\alpha^T\cdots\times_K\widehat{\mathbf{p}}^T\\ =& \mathcal{C}_\times\times_1\widehat{\mathbf{p}}^T\times_2\widehat{\mathbf{p}}^T\cdots\times_{K-1}\widehat{\mathbf{p}}^T\times_K\mathbf{e}_\alpha^T\\ =& G(\overbrace{\widehat{\mathbf{p}},\cdots,\widehat{\mathbf{p}}}^{K-1 \text{ players}},\mathbf{e}_\alpha)\end{aligned} \tag{5.18}$$

Substituting (5.18) into (5.17), we have the following relations

$$\begin{aligned}\lambda &= u_1 + K\sum_{\alpha_2=1}^{N_\times}\cdots\sum_{\alpha_K=1}^{N_\times} C_{1,\alpha_2,\cdots,\alpha_K}\prod_{n=2}^{K}\widehat{p}_{\alpha_n},\\ \lambda &= u_2 + K\sum_{\alpha_2=1}^{N_\times}\cdots\sum_{\alpha_K=1}^{N_\times} C_{2,\alpha_2,\cdots,\alpha_K}\prod_{n=2}^{K}\widehat{p}_{\alpha_n},\\ &\vdots\\ \lambda &= u_\alpha + K\sum_{\alpha_2=1}^{N_\times}\cdots\sum_{\alpha_K=1}^{N_\times} C_{\alpha,\alpha_2,\cdots,\alpha_K}\prod_{n=2}^{K}\widehat{p}_{\alpha_n},\\ &\vdots\\ \lambda &= u_{N_\times} + K\sum_{\alpha_2=1}^{N_\times}\cdots\sum_{\alpha_K=1}^{N_\times} C_{N_\times,\alpha_2,\cdots,\alpha_K}\prod_{n=2}^{K}\widehat{p}_{\alpha_n}.\end{aligned} \tag{5.19}$$

Let $\mathbf{e}$ be an $N_\times$-dimensional all one vector $\mathbf{e}=[1,\cdots,1]^T$ and $\mathbf{z}=[z_1,z_2,\cdots,z_{N_\times}]^T$ with the αth element $z_\alpha=\sum_{\alpha_2=1}^{N_\times}\cdots\sum_{\alpha_K=1}^{N_\times}C_{\alpha,\alpha_2,\cdots,\alpha_K}\prod_{n=2}^{K}\widehat{p}_{\alpha_n}$. We have the relations in (5.19) rewritten in terms of matrices as follows

$$\lambda\mathbf{e}=\mathbf{u}+K\mathbf{z}. \tag{5.20}$$

Performing inner product on both sides of (5.20) with $\widehat{\mathbf{p}}$, we have

$$\lambda\mathbf{e}\cdot\widehat{\mathbf{p}}=\mathbf{u}\cdot\widehat{\mathbf{p}}+K\mathbf{z}\cdot\widehat{\mathbf{p}}. \tag{5.21}$$

Since $\mathbf{e}\cdot\widehat{\mathbf{p}}=1$ and $K\mathbf{z}\cdot\widehat{\mathbf{p}}=\sum_{\alpha_1=1}^{N_\times}\cdots\sum_{\alpha_K=1}^{N_\times}C_{\alpha_1,\cdots,\alpha_K}\prod_{n=1}^{K}\widehat{p}_{\alpha_n}$, we have

$$\lambda=\sum_{\alpha=1}^{N_\times}u_\alpha\widehat{p}_\alpha+\sum_{\alpha_1=1}^{N_\times}\cdots\sum_{\alpha_K=1}^{N_\times}C_{\alpha_1,\cdots,\alpha_K}\prod_{n=1}^{K}\widehat{p}_{\alpha_n}. \tag{5.22}$$

If for $1\le\alpha\le N_\times$, $u_\alpha\ge 0$ and $\sum_{\alpha=1}^{N_\times}u_\alpha p_\alpha=0$, the vector $\mathbf{p}$ with the αth element p_α is referred to as a point satisfying the Karush-Kuhn-Tucker (KKT) conditions [56] for the Lagrange function (5.16). Therefore, for the optimal matching probability vector $\widehat{\mathbf{p}}$ which satisfies KKT conditions, $\lambda=\sum_{\alpha_1=1}^{N_\times}\cdots\sum_{\alpha_K=1}^{N_\times}C_{\alpha_1,\cdots,\alpha_K}\prod_{n=1}^{K}\widehat{p}_{\alpha_n}=G(\overbrace{\widehat{\mathbf{p}},\cdots,\widehat{\mathbf{p}}}^{K\text{ players}})$ is the maximum value of the average payoff (5.15). In this case, the relation (5.17) can be rewritten as

$$G(\overbrace{\widehat{\mathbf{p}},\cdots,\widehat{\mathbf{p}}}^{K-1\text{ players}},\mathbf{e}_\alpha)+u_\alpha\widehat{p}_\alpha=G(\overbrace{\widehat{\mathbf{p}},\cdots,\widehat{\mathbf{p}}}^{K\text{ players}}) \tag{5.23}$$

According to the KKT conditions, $u_\alpha=0$ if $p_\alpha>0$. Thus we have the following two relations

$$G(\overbrace{\widehat{\mathbf{p}},\cdots,\widehat{\mathbf{p}}}^{K-1\text{ players}},\mathbf{e}_\alpha)\le G(\overbrace{\widehat{\mathbf{p}},\cdots,\widehat{\mathbf{p}}}^{K\text{ players}}),\quad\text{for } 1\le\alpha\le N_\times; \tag{5.24}$$

$$G(\overbrace{\widehat{\mathbf{p}},\cdots,\widehat{\mathbf{p}}}^{K-1\text{ players}},\mathbf{e}_\alpha)=G(\overbrace{\widehat{\mathbf{p}},\cdots,\widehat{\mathbf{p}}}^{K\text{ players}}),\quad\text{for } \alpha \text{ with respect to } p_\alpha>0. \tag{5.25}$$

The relations in (5.24) and (5.25) are in fact sufficient conditions for a Nash equilibrium of an $N_\times$-strategy, K-player evolutionary game [13]. We refer to $G(\overbrace{\widehat{\mathbf{p}},\cdots,\widehat{\mathbf{p}}}^{K-1\text{ players}},\mathbf{e}_\alpha)$ as the average payoff obtained by the α-strategist. Here an α-strategist is a player who insists on choosing the strategy α in the process regardless of the evolution of the remaining strategies. This is in contrast to the overall average payoff $G(\overbrace{\widehat{\mathbf{p}},\cdots,\widehat{\mathbf{p}}}^{K\text{ players}})$ where the probability $\widehat{\mathbf{p}}$ of the evolutionarily stable strategies can be achieved through the evaluation among all the $N_\times$ strategies. We can thus alternatively interpret the optimization of the matching probability vector $\widehat{\mathbf{p}}$ as a Nash equilibrium in the game with compatibility $\mathcal{C}_\times$, subject to the KKT conditions.

5.5 Experiments

We test our algorithm for high order structural matching on two types of data. Firstly, we test our method on synthetic data to evaluate its robustness to noise and outliers. Secondly, we conduct experiments to match features extracted from images. Prior rejections are considered for both types of data to improve the matching accuracy. We compare our method with two state-of-the-art methods, i.e. probabilistic hypergraph matching [110] and tensor power iteration [28].

5.5.1 Matching Synthetic Data

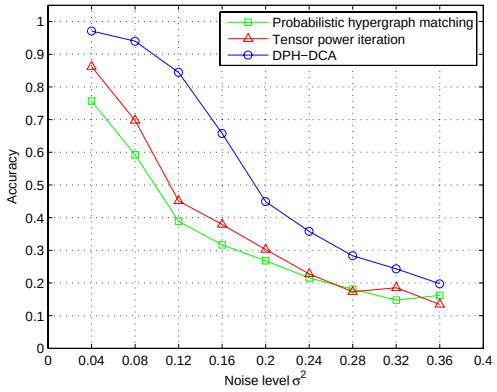
We commence with the random generation of a structural prototype with 15 vertices. The distance d_{ij} between each pair of vertices i and j of the prototype is randomly distributed subject to the Gaussian distribution $N(1, 0.5)$. We test our method by establishing correspondence between the prototype structure and a modified structure. The alternative modifications include a) noise addition, b) vertex deletion, c) rescaling and d) rotation. Since neither the probabilistic hypergraph matching nor tensor power iteration methods rely on a specific initialization, we test our DPH-DCA matching method without prior rejections to make a fair comparison with these two alternative methods.

We first compare the performance of different methods under a varying degree of added noise, both for graph matching and hypergraph matching. To this end we perturb the prototype structure by adding noise to the distance between each vertex pair and generate a modified structure. The noise added is normally distributed according to $N(0, \sigma^2)$, and we vary σ^2 from 0.04 up to 0.36 to evaluate the matching accuracy at different noise levels. We conduct 100 trials at each noise level. For graph matching, we measure the pairwise similarity between the two points i and j within each set by using $g_{ij} = \exp(-\|d_{ij}\|^2/\sigma_2)$ where σ_2 is the scaling parameter and is set to 0.5 in this experiment. One the other hand, when pairwise measures can not properly reflect the

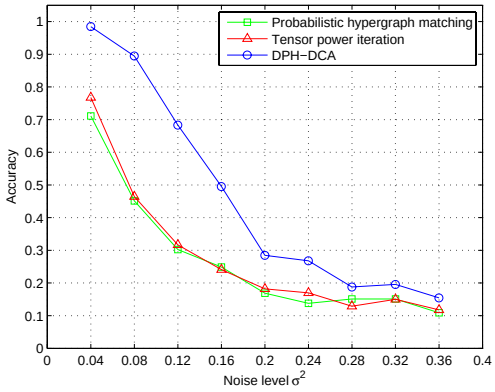
similarities between structures, e.g. similar polygons with different areas, higher order similarity measures are required for establishing hypergraph representations. To test the performance of different methods for hypergraph matching we re-scaled the distance between each of vertex pairs by a random factor and rotate the structure by a random angle. In this case, the pairwise relationships are no longer sufficient for the matching task. We use the sum of polar sines presented in [53] as a higher order similarity measure for point tuples. We measure the similarity of every 3-tuple within the vertex set and thus establish a weighted 3-uniform hypergraph for the structure. The compatibility tensor $\mathcal{C}$ for two structures is computed according to (5.1) with $\sigma_1 = 0.1$. Figures 5.10(a) and 5.10(b) illustrate the results of the matching accuracy as a function of noise level. It is clear that our DPH-DCA framework outperforms the two alternative methods at each noise level.

To take the investigation one step further, we study the performance of our method for matching structures of different vertex cardinalities. To this end, we extract a substructure from a prototype and slightly perturb the distance between each vertex pair by adding random noise normally distributed according to $N(0, 0.04)$. The cardinality of the vertex set of the substructure varies from 14 down to 5. Vertices not in the substructure are outliers for the matching process. For each vertex cardinality of a substructure, 100 trials are performed. Figures 5.10(c) and 5.10(d) illustrate the matching accuracy as a function of outlier number for the three methods. It is clear that our DPH-DCA framework outperforms the two alternative methods at each number of outliers.

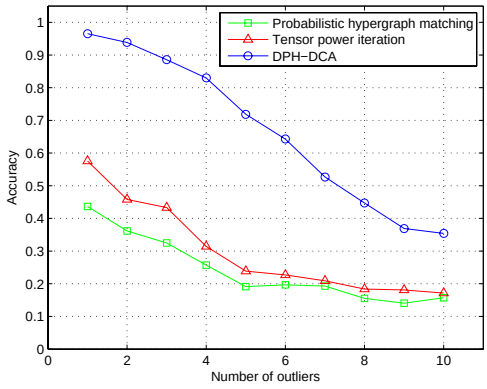
We have also evaluated the matching accuracy of our DPH-DCA framework at different levels of available prior rejections. To this end, we have extracted a 5-vertex substructure from a prototype and slightly perturb the distance between each vertex pair by adding random noise normally distributed according to $N(0, 0.04)$. We involve prior rejections by rejecting the matchings associated with a varying number of outliers. Figures 5.11(a) and 5.11(b) illustrate the matching accuracy as a function of the number of rejected outliers. It is clear that the matching accuracy grows monotonically as the number of rejected



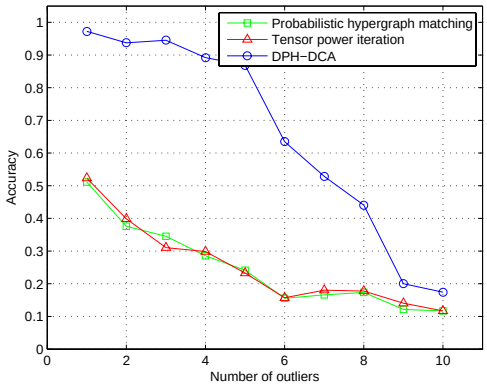
(a) Graphs.



(b) Hypergraphs.

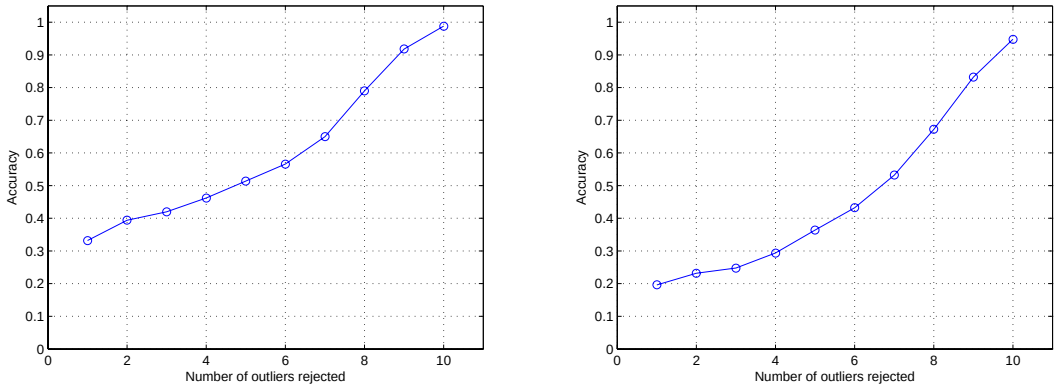


(c) Graphs.



(d) Hypergraphs.

FIGURE 5.10: MATCHING PERFORMANCE AT DIFFERENT LEVELS OF NOISE AND DIFFERENT NUMBERS OF OUTLIERS.



(a) Graphs. (b) Hypergraphs.

FIGURE 5.11: IMPROVEMENT OF MATCHING PERFORMANCE WITH PRIOR REJECTIONS.

outliers increases.

5.5.2 Image Correspondences

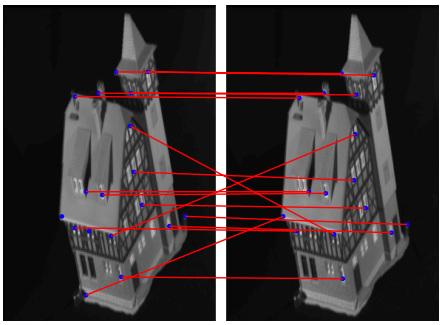
We apply the different methods to the CMU house sequence, which has been widely used to test matching algorithms [16][21][28][57] and provides a baseline for comparison. We perform graph matching for this dataset and evaluate the performance for alternative methods. We use the Harris detector [39] to extract corner points from the images. The matching results for the first and tenth frames in the sequence are shown in Figure 5.12. The probabilistic hypergraph matching produces three incorrect correspondences, tensor power iteration produces four incorrect correspondences, and our DPH-DCA algorithm (without prior rejections) establishes all correct matchings. The matching results for the first and twentieth frames in the sequence are shown in Figure 5.13. From Figures 5.13(a), 5.13(b) and 5.13(c) it is clear that although our DPH-DCA algorithm outperforms the two alternative methods, it still produces two false matches. The reasons for this is two-fold. Firstly, relative to the corner points extracted in the first frame, the positional disturbance

on the corner points in the twentieth frame is greater than that on those in the tenth frame. Secondly and more importantly, there are three corner points in the bottom part of the first frame which have no exact correspondence with any points in the twentieth frame, and thus can be considered as outliers. The positional disturbance and outliers are a consequence of the limited effectiveness of the corner detector for extracting invariant features from frames where the noise variance is significant. Figure 5.13(d) shows the correspondences produced by our DPH-DCA algorithm with prior rejections and the green marked points are the those rejected as outliers. It is clear that after prior rejections, all of the correspondences produced by DPH-DCA are visually correct though there are still positional disturbances for the corner points.

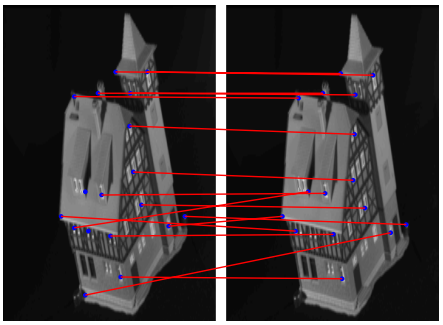
We then apply the alternative matching methods to an image sequence of a toy human being, which has been used in the experiments of the preceding chapters for structural clustering. In Figure 5.14 we visualize the performance of the alternative methods by matching the first and twelfth frames. Furthermore, we test the alternative methods to match a pair of images from one class of COIL dataset and the matching results are illustrated in Figure 5.15. It is clear that DPH-DCA provides the most visually correct correspondences among alternative methods.

To make a quantitative evaluation of the alternative methods, we plot the matching accuracy between the first frame and the subsequent frames in an image sequence as a function of frame number. The accuracy for DPH-DCA is computed without prior rejections. The accuracy curves for the image sequences of the CMU house and the toy human being are shown in Figures 5.16(a) and 5.16(b), respectively. From all the experimental results for the two sequences of images, it is clear that our DPH-DCA method (without rejections) outperforms the two alternative methods. Moreover, with the assistance of prior rejection, we can further improve the matching results of DPH-DCA.

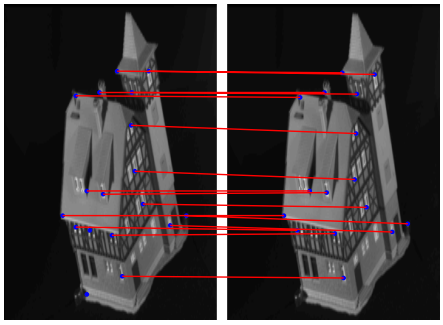
To visualize the matching for real world images we test the alternative methods on



(a) Probabilistic hypergraph matching.

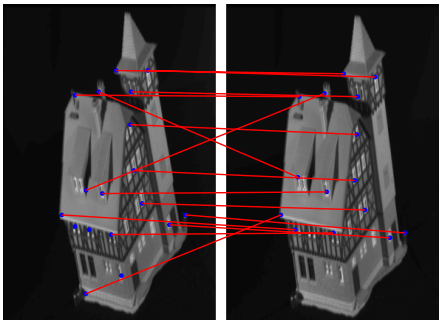


(b) Tensor power iteration.

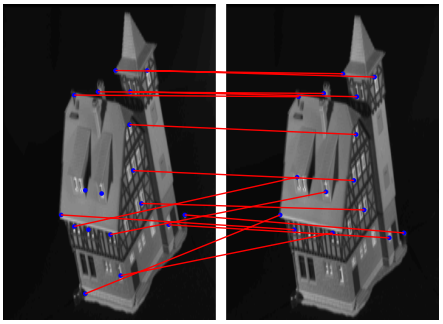


(c) DPH-DCA.

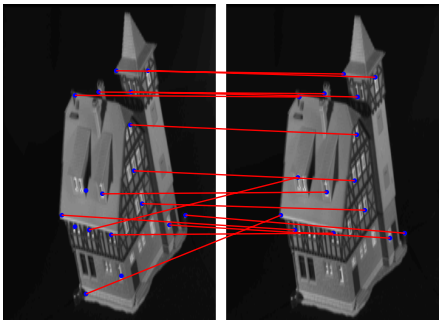
FIGURE 5.12: CORRESPONDENCE BETWEEN THE FIRST AND TENTH HOUSE IMAGES.



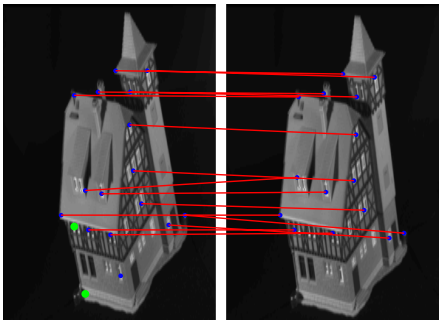
(a) Probabilistic hypergraph matching.



(b) Tensor power iteration.

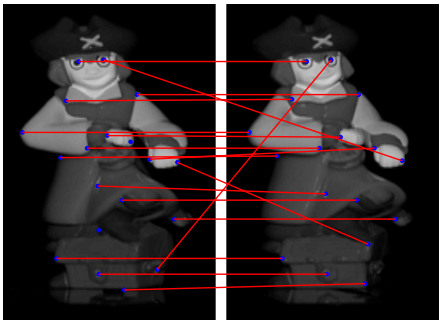


(c) DPH-DCA.

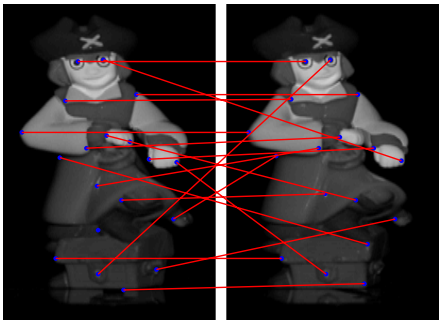


(d) DPH-DCA with prior rejections.

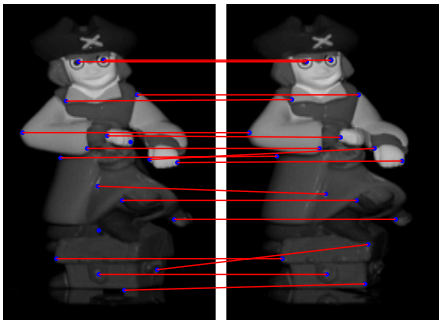
FIGURE 5.13: CORRESPONDENCE BETWEEN THE FIRST AND TWENTIETH HOUSE IMAGES.



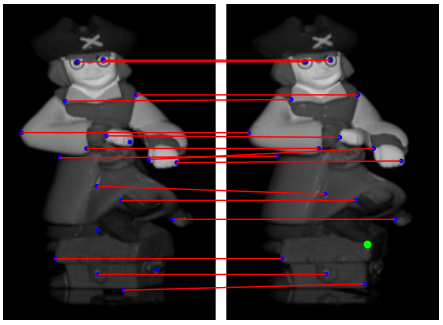
(a) Probabilistic hypergraph matching.



(b) Tensor power iteration.

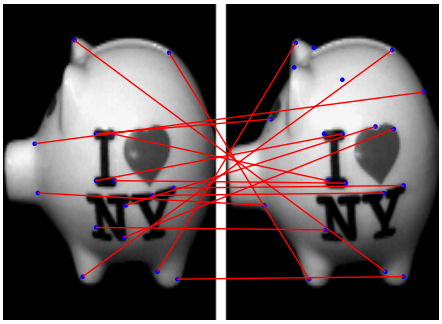


(c) DPH-DCA.

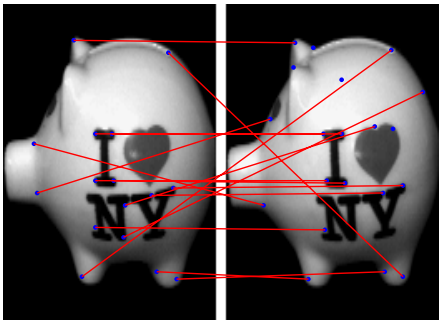


(d) DPH-DCA with prior rejections.

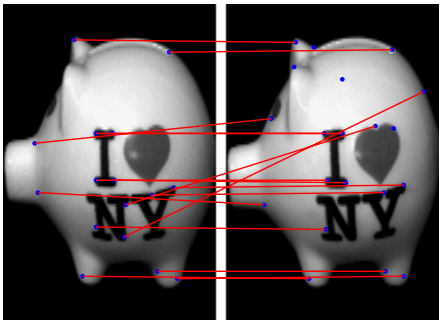
FIGURE 5.14: CORRESPONDENCE BETWEEN THE MODEL AND TWELFTH IMAGES OF THE TOY HUMAN BEING.



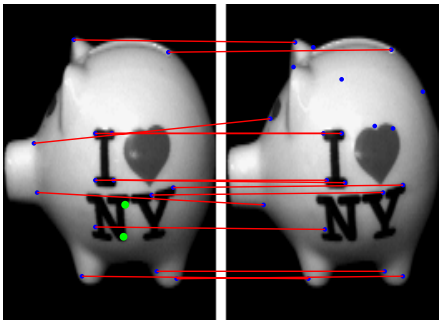
(a) Probabilistic hypergraph matching.



(b) Tensor power iteration.

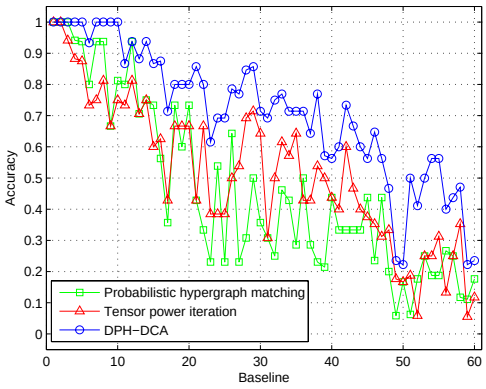


(c) DPH-DCA.

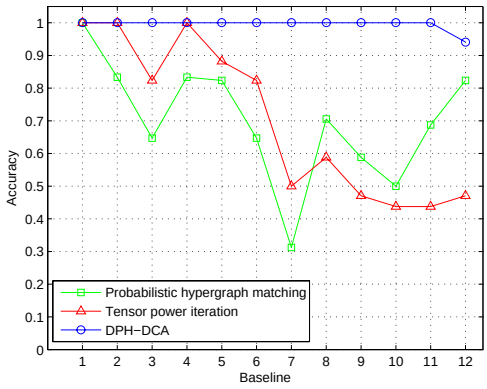


(d) DPH-DCA with prior rejections.

FIGURE 5.15: CORRESPONDENCE BETWEEN THE IMAGES OF THE TOY PIGS.



(a) Images for the CMU house.



(b) Images for the toy human being.

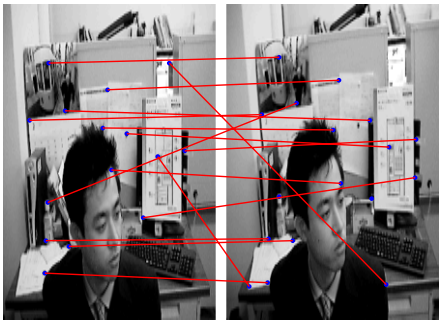
FIGURE 5.16: GRAPH MATCHING PERFORMANCE FOR IMAGES.

frames of video[1]. We use the Harris detector to extract corner points from the first and 30th frames. We use the sum of polar sines presented in [53] to measure the similarity of every 3-tuple in the corner point set and thus establish a weighted 3-uniform hypergraph for each image. Figure 5.17 illustrates the matching performances for alternative methods. The matching results for the two comparison methods are visualized in Figures 5.17(a) and 5.17(b), where 11 correct correspondences and 4 incorrect ones are obtained by using the tensor power iteration, and 12 correct correspondences and 3 incorrect ones by the probabilistic hypergraph matching. For DCA without prior rejections (visualized in Figure 5.17(c)), we obtain 14 correct correspondences and 1 incorrect ones. Figure 5.17(d) visualizes the matching result by rejecting two outliers (green marked). It is clear that the false match is eliminated by incorporating the proper prior rejections.

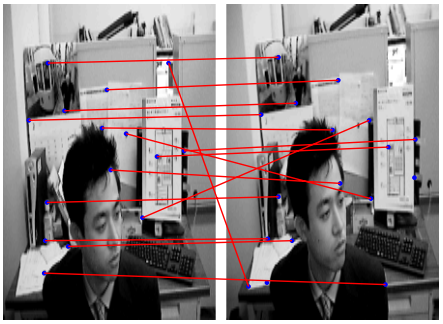
5.5.3 Test for Nash Equilibrium

In Section 5.4 we have justified the DPH-DCA framework in terms of evolutionary game theory. Furthermore, we have also observed that the optimal solution obtained by DPH-DCA achieves a Nash equilibrium subject to the Karush-Kuhn-Tucker (KKT) conditions. To experimentally verify this observation, we make a quantitative evaluation on the average payoff obtained by evolutionarily stable strategies both with and without an α-strategist. We first compute the average payoff for the matchings demonstrated in Figures 5.12(c), 5.13(c), 5.14(c) and 5.15(c) by using equation (5.15). The values for the overall average payoff (i.e. that without an α-strategist) are plotted as red asterisks in the four subfigures of Figure 5.18. Every unmatched pair of vertices represents a strategy α which has been driven to extinction in the evolution. For the α-strategist selecting the extinct strategy α, we compute the average payoff with respect to $\widehat{\mathbf{p}}$ and $\mathbf{e}_\alpha$ by using equation (5.18). The sorted average payoff involving extinct strategies (i.e. unmatched vertex pairs) are plotted as black triangles in the four subfigures of Figure 5.18. It is clear that

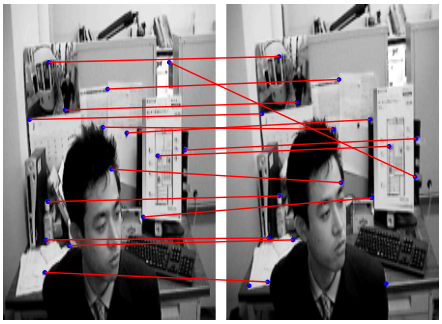
[1]http://www.suri.it.okayama-u.ac.jp/e-program-separate.html



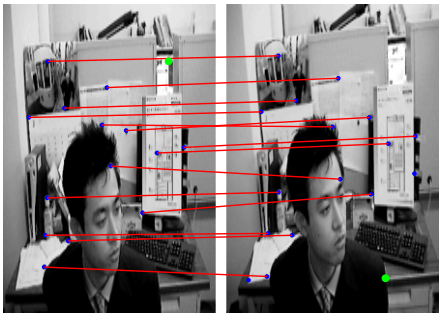
(a) Probabilistic hypergraph matching.



(b) Tensor power iteration.



(c) DPH-DCA.



(d) DPH-DCA with two prior rejections.

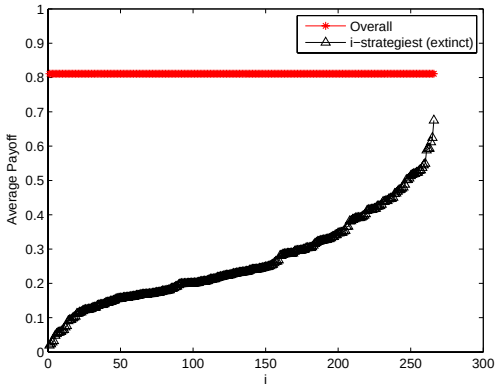
FIGURE 5.17: IMAGE CORRESPONDENCES.

the values of the average payoff obtained with consideration of an α-strategist are less than those of the overall average payoff.

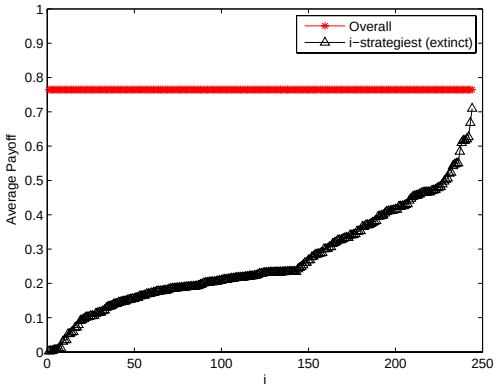
We then evaluate the average payoff for all the matching pairs in the image sequences of the CMU house and the toy human being. For each two images to be matched, we refer to the pair of the leftmost vertex in one image and the rightmost vertex in the other image as the extinct strategy α. This assumption is reasonable because these two vertices always tend to be so different in structural context that they are very unlikely (almost impossible) to be matched by using alternative matching methods. We plot the overall average payoff for each pair of images as red asterisks and plot the average payoff with the α-strategist as black triangles. It is easy to observe that the values of the overall average payoff overwhelm those with the α-strategist. Both Figures 5.18 and 5.19 experimentally verify the fact that the high order matching satisfies the relation (5.25) in Nash equilibrium, and thus provide a practical support for our interpretation in Section 5.4.

5.6 Summary

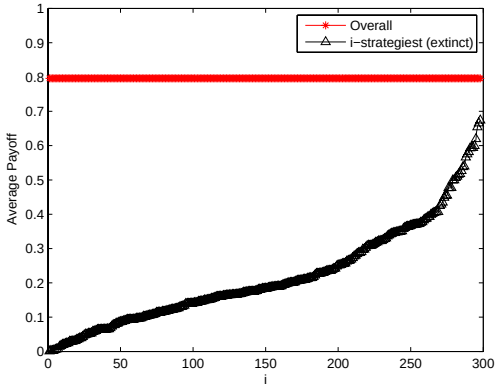
In this chapter we have presented a novel approach to high order structural matching. We have transformed the matching problem to that of extracting the dominant cluster from the direct product hypergraph for two feature sets with high order relationships. Prior knowledge about outliers can be easily involved in our framework by initializing the matchings associated with the outliers by a zero weight. Additionally, we have explained how to justify the proposed framework in terms of evolutionary game theory. Experiments have shown that our method outperforms the state-of-the-art methods.



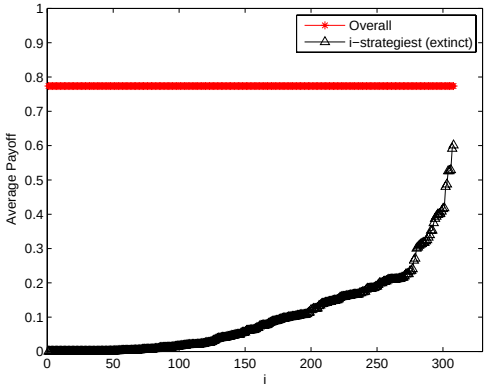
(a) The first and tenth house frames.



(b) The first and twentieth house frames.

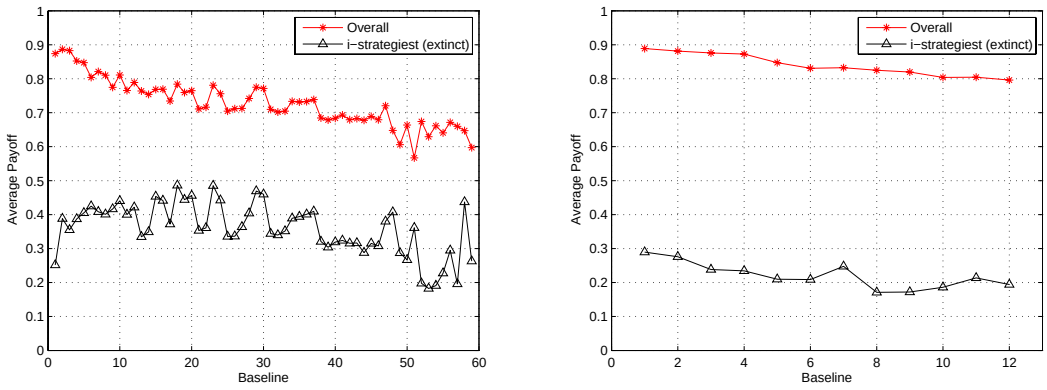


(c) Images for the model toy human being and the twelfth toy human being.



(d) Images for toy pigs.

FIGURE 5.18: AVERAGE PAYOFF ASSOCIATED WITH CORRESPONDENCES FOR A PAIR OF IMAGES.



(a) House image sequence.

(b) Image sequence for toy human being.

FIGURE 5.19: AVERAGE PAYOFF ASSOCIATED WITH CORRESPONDENCES FOR DIFFERENT IMAGE PAIRS.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

We have presented two structural learning methods, Ihara coefficients for the purpose of structural characterization and DPH-DCA for the purpose of structural matching. These two methods can be applied to both pairwise and higher order relational structures.

To develop an effective vectorial representation for graphs, we have firstly studied how to extract characteristics from graphs using the Ihara zeta function. We have used a set of polynomial coefficients, i.e. Ihara coefficients, derived from a transformed graph to construct pattern vectors. Furthermore, we have provided a route that enables the Ihara coefficients to be extended from unweighted to edge-weighted graphs. This is accomplished by establishing the Perron-Frobenius operator for edge-weighted graphs with the assistance of a reduced Barthodi zeta function. This generalization allows us to compute the Ihara coefficients for both unweighted graphs and edge-weighted graphs in a unifying manner. We have performed a spectral analysis that explains the reasons for the Ihara coefficients having greater representational power than the graph spectral methods. We have also shown how the Ihara coefficients can be computed in terms of the spectrum of a transformed graph. Additionally, the Ihara coefficients are by definition closely related to

graph topologies. We can thus conclude that the Ihara coefficients are polynomial characteristics combining both topological properties and spectral features. This advantage makes the Ihara coefficients a more powerful graph characterization method over those restricted to either topologies or spectra.

We have also performed a polynomial analysis on hypergraphs and characterize (nonuniform) unweighted hypergraphs based on the Ihara zeta function. We have used the polynomial coefficients, i.e. Ihara coefficients, as the elements in the pattern vector for a hypergraph. We have verified that the Ihara coefficients are capable of distinguishing relational orders and thus avoid the ambiguities caused by the hypergraph Laplacian. Although the hypergraph Ihara coefficients are characteristics extracted in a polynomial manner, they can be computed in terms of the adjacency spectrum of a colored oriented line graph representation of the original hypergraph. Additionally, the definition of the hypergraph Ihara zeta function enables the coefficients to reflect certain topological properties of the associated colored oriented line graph. Therefore, the Ihara coefficients can be regarded as a set of quite general characteristics, as they bear information regarding polynomials, spectra and topologies. Furthermore, we have also introduced an efficient method for computing the hypergraph Ihara coefficients based on the associated bipartite graph. This is effected due to the fact that the associated bipartite graph results in a relatively small sized graph representation of the hypergraph, on which the Ihara coefficients are computed. All the above merits make the Ihara coefficients a flexible and effective method for high order structural characterization.

In the last part of this thesis we have presented a novel approach to high order structural matching. We have cast the problem of high order matching to that of high order clustering. To this end, we have first established a direct product hypergraph for two feature sets with high order relationships. In the direct product hypergraph, every vertex represents a possible correspondence between a pair of vertices drawn separately from the two feature sets. The vertices representing correct correspondences are supposed to form

a dominant cluster and those for incorrect correspondences should not be included in the cluster. Thus, we have developed a method for extracting the dominant cluster for correct correspondences, which is referred to as *dominant cluster analysis* on a *direct product hypergraph* (DPH-DCA). We have also shown that prior knowledge about outliers in either feature set can be easily incorporated into our framework by initializing the matchings associated with the outliers by a zero weight. Our DPH-DCA framework naturally satisfies the basic axioms of probability, i.e. sum to one and positiveness. Furthermore, we have justified our framework in terms of evolutionary game theory. We have observed that a Nash equilibrium can be achieved in our matching framework subject to the KKT conditions. Last but not least, our matching approach is a flexible method that can be applied to establishing correspondences between either a pair of graphs or a pair of hypergraphs.

We have conducted experiments on both pairwise and higher order structured data using the proposed methods. For each method, we have made an experimental comparison with state of the art methods, where both qualitative and quantitative evaluations have been made. Experimental results show the effectiveness of the method presented in this thesis.

6.2 Limitations

Although the methods described in this thesis outperform the state of the art methods, there are still some limitations to be noted.

We have shown both theoretically and experimentally that the Ihara coefficients outperform the spectral methods in graph characterization. However, the Ihara coefficients do not apply to hierarchical structures, e.g. ordered trees. As many practical problems exhibit themselves with hierarchical structures, the Ihara coefficients see limits in the applications of analyzing tree-like graphs, such as molecules and internet. The reason for this limitation is that the Ihara zeta function is governed by cycle frequencies of graphs

and the Ihara coefficients are thus insufficient to reflect structural characteristics such as branches and vertex orderings. As a result, the Ihara coefficients are not capable of distinguishing graphs with the same cyclic structure but different vertex orderings or branch structures.

Although we have described how to characterize hypergraphs using a high order generalization of the Ihara coefficients, this representation is restricted to unweighted hypergraphs. On the other hand, edge-weighted hypergraph representations, which are capable of not only capturing the multiple connections among vertices, but also depicting the affinity of the vertices encompassed in a hyperedge, have emerged in the recent literature. The hypergraph Ihara coefficients described in this thesis, however, do not exhibit themselves with the capability of handling edge-weighted hypergraphs. In the representation of a hypergraph using the colored oriented line graph, edges drawn from the same hyperedge share the common color. As far as edge-weighted hypergraphs are concerned, not only is the same color required for edges drawn from a common hyperedge but also the weight of each hyperedge is supposed to be broken down and assigned to its derivative edges according to certain criteria. Since there has not yet been a widely accepted method for graph representations of hypergraphs, the extension of hypergraph Ihara coefficients to edge-weighted hypergraphs is still a problem to be solved.

Additionally, the feature selection method adopted for (hyper)graph characterization based on Ihara coefficients is heuristic. The individual Ihara coefficients contribute in quite different ways to (hyper)graph characterization. Furthermore, their different combinations manifest a variation of the representational power of the coefficients. It is thus important to understand their individual significance and their optimal combinations.

In the general matching scenario, the similarity measure between hyperedges plays an important role in establishing the compatibility tensor and has a great influence on the subsequent matching performance. In this thesis, we have used the Gaussian kernel for measuring the similarity of a pair of hyperedges, and we have used the sum of polar sines

as a higher order similarity measure for point tuples. Although these measures have already been used in algorithms for various pattern recognition problems, there is still no theoretical evidence to prove them to be optimal options. Therefore, the choice of similarity measures in this work is heuristic, and we need to carry out a further investigation on how to define a reasonable similarity measure that is capable of reflecting structural features more convincingly.

6.3 Future Work

To address the shortcomings described in the preceding section, we suggest some possible approaches to overcoming them in future work.

To generalize the functionality of the Ihara coefficients and make them available to hierarchical structures such as ordered trees, we may develop a new zeta function with the consideration of N-step backtracking. In the new definition, the zeta function is expected to be refined such that its polynomial coefficients are not only suitable to distinguish branch structures but also capable of incorporating vertex ordering information.

K-uniform hypergraphs with weighted hyperedges have recently been exploited as a powerful tool in representing data with high order relations. It is thus worth trying to generalize the Ihara zeta function to admit edge-weighted hypergraphs. To this end, we have to determine the criteria upon which a reasonable edge-weighting strategy for graph representations of hypergraphs can be established. Furthermore, we need to work out the method for computing the solution that satisfies the criteria.

Since the method of feature selection plays an important part in determining the representational power of the Ihara coefficients, it might be interesting for us to adopt some more sophisticated strategies (e.g. simulated annealing) to identify the most discriminative coefficient subset.

For our hypergraph matching framework, we need to develop more accurate similar-

ity measures by investigating more sophisticated metrics that can effectively capture the relationship between hyperedges. We also need to develop strategies for evaluating the effectiveness of varying parameter values in the similarity measure and thus obtaining the optimal solution. Other possible future work regarding DPH-DCA includes developing the combined methodology for both high order clustering and matching. As the DPH-DCA method formulates the problem of matching in terms of clustering, it should be possible to establish a unifying framework both for high order clustering and matching based on DPH-DCA.

Furthermore, the methodologies developed in the thesis should not be confined to computer vision. Possible applications can be explored in various research fields such as biological networks, molecular computing, social networks and complex systems.

Appendix A

Relationship between the Ihara Zeta Function and Discrete-Time Quantum Walks

A.1 Discrete-time Quantum Walks

This section reviews the fundamental knowledge of discrete-time quantum walks. These notions are derived from the study of a quantum mechanical system. Detailed knowledge of quantum mechanics reviewed in this section can be found in the textbook [81] and the thesis [29].

To establish discrete-time quantum walks on a graph $G(V, E)$, we first replace each edge $e(u, v) \in E$ of the graph by a pair of reverse arcs $e_d(u, v)$ and $e_d(v, u)$, and denote the set of arcs by E_d, which is also referred to as the state space for the discrete-time quantum walks. Using Dirac's notation, we denote the state on the arc $e_d(u, v)$ by $|uv\rangle$, and the general state of the quantum walks is of the form

$$|\psi\rangle = \sum_{e_d(u,v)\in E_d} \alpha_{uv}|uv\rangle, \tag{A.1}$$

where the quantum amplitudes α_{uv} are complex. Using (A.1), the probability that the walk is in the state $|uv\rangle$ is given by $\Pr(|uv\rangle) = \alpha_{uv}\alpha^*_{uv}$, where α^*_{uv} is the complex conjugate of α_{uv}.

The evolution of the state vector between the steps t and $t+1$ is determined by the transition matrix $\boldsymbol{U}$. The entries of $\boldsymbol{U}$ determine the probabilities for transitions between states, i.e. $|\psi_{t+1}\rangle = \boldsymbol{U}|\psi_t\rangle$. Since the evolution of the walk is linear and conserves probability, the matrix $\boldsymbol{U}$ must be unitary, i.e. $\boldsymbol{U}^{-1} = \boldsymbol{U}^\dagger$, where $\boldsymbol{U}^\dagger$ denotes the complex conjugate of the matrix transposed. Furthermore, the transition matrix $\boldsymbol{U}$ is supposed to assign the same amplitudes to all transitions $|u_1 v\rangle \rightarrow |v u_i\rangle$, $u_i \in \mathcal{N}(v) \setminus u_1$ ($2 \leq i \leq r$) and a different amplitude to the transition $|u_1 v\rangle \rightarrow |v u_1\rangle$, because the walks on an unweighted graph does not rely on any labeling of the edges or vertices. The Grover diffusion matrices [37] are usually adopted as the transition matrices, because they are the matrices furthest from the identity which are unitary and are not dependent on any labeling of the vertices. Using the Grover diffusion matrices, the transition matrix $\boldsymbol{U} = [U_{(w,x),(u,v)}]_{e_d(w,x),e_d(u,v)\in E_d}$ has entries

$$U_{(w,x),(u,v)} = \begin{cases} \frac{2}{d_v} - \delta_{ux}, & v = w, \\ 0, & \text{otherwise}, \end{cases} \tag{A.2}$$

where δ_{ux} is Kronecker delta, i.e. $\delta_{ux} = 1$ if $u = x$ and 0 otherwise. The definition in (A.2) allows *destructive interference* to take place, because the entries of $\boldsymbol{U}$ are negative as well as positive, though be real. The negative entries of $\boldsymbol{U}$ can result in negative quantum amplitudes for a state.

A.2 Relationship between the Ihara Zeta Function and Discrete-time Quantum Walks

In this section we will show how to express the Perron-Frobenius operator in terms of the transition matrix of the discrete-time quantum walks. To achieve this goal, we introduce the definition of the positive support of a matrix.

Definition 3. *The positive support* $\mathrm{S}^{+}(\mathbf{M}) = [s_{i,j}]_{m\times n}$ *of the matrix* $\mathbf{M} = [M_{i,j}]_{m\times n}$ *is defined to be a matrix with entries*

$$s_{i,j} = \begin{cases} 1, & M_{i,j} > 0, \\ 0, & \text{otherwise}, \end{cases} \tag{A.3}$$

where $1 \leq i \leq m, 1 \leq j \leq n$ *.*

Theorem 2. *Let* ***T*** *be the Perron-Frobenius operator in the Ihara zeta function of a simple graph* $G(V, E)$ *subject to the md2 constraint (i.e. that any vertex of a simple graph are adjacent to at least two other vertices). Let* ***U*** *be the Grover diffusion matrix governing the evolution of the discrete-time quantum walks on the graph* $G(V, E)$*. Then* ***T*** *is the positive support of the transpose of* ***U****, i.e.* $\boldsymbol{T} = \mathrm{S}^{+}(\boldsymbol{U}')$.

Proof. For the graph $G(V, E)$, the state transition matrix ***U*** of the discrete-time quantum walks and the Perron-Frobenius operator ***T*** of the Ihara zeta function are both $2|E| \times 2|E|$ matrices. This is because both the cardinality $|V_L|$ of the vertex set V_L of the associated oriented line graph $OLG(V_L, E_{dL})$ and the total number of basis states in the discrete-time quantum walks are equal to the cardinality $|E_d|$ of the arc set E_d of the associated symmetric digraph $SDG(V, E_d)$, and it is obvious that $|E_d| = 2|E|$.

Specifically, all the non-zero entries of ***T*** are 1 while the same entries in $\boldsymbol{U}'$ are weighted by twice of the reciprocal of the connecting vertex degree in the graph $G(V, E)$. Additionally, the entries representing reverse arcs in $\boldsymbol{U}'$ have values $2/d_v - 1$ $(v \in V)$ while the same entries in ***T*** are always set to zeros.

The md2 constraint requires any vertex of the simple graph $G(V, E)$ are adjacent to at least two other vertices ($d_v \geq 2$, $v \in V$). Therefore,

$$\frac{2}{d_v} - 1 \leq 0. \tag{A.4}$$

We thus have

$$\begin{aligned} &U_{(w,x),(u,v)} > 0 \ \text{ if } \ v = w \text{ and } u \neq x, \\ &U_{(w,x),(u,v)} \leq 0 \ \text{ otherwise}, \end{aligned} \tag{A.5}$$

for $u, v, w, x \in V$ and $e_d(w, x), e_d(u, v) \in E_d$. From the relations in (A.5) we can conclude that $\boldsymbol{T} = \mathrm{S}^+(\boldsymbol{U}')$. This completes the proof. □

A.3 The Ihara Zeta Function and Cospectral Regular Graphs

Scott and Storm [79] have numerically proved that the Ihara zeta function is superior to the adjacency matrix spectrum in distinguishing non-isomorphic graphs. Furthermore, they have pointed out that in some circumstances the Ihara zeta function can not distinguish non-isomorphic graphs. However, they have not offered an explanation or analysis of the cases in which the Ihara zeta function fails to resolve spectral ambiguities. From our empirical work [30][31] and based on the relationship between the Ihara zeta function and discrete-time quantum walks described in Section A.2, we assert that the Ihara zeta function is not able to properly distinguish cospectral regular graphs.

To verify this assertion, we first establish the matrix $\tilde{\boldsymbol{T}} = [\tilde{T}_{(u,v),(w,x)}]_{u,v,w,x \in V}$ on the graph $G(V, E)$. The entries of $\tilde{\boldsymbol{T}}$ are

$$\tilde{T}_{(u,v),(w,x)} = A_{uv} A_{wx} \delta_{vw} \Big(1 - \delta_{ux} \Big), \tag{A.6}$$

where $u, v, w, x \in V$ and A_{uv} is the (u, v)th entry of the adjacency matrix of $G(V, E)$. Consequently, $\tilde{\boldsymbol{T}}$ is a $|V|^2 \times |V|^2$ matrix which can be established in a similar manner to the Perron-Frobenius operator $\boldsymbol{T}$. It is the adjacency matrix of a transformed graph $G_{\tilde{\boldsymbol{T}}}(V_{\tilde{\boldsymbol{T}}}, E_{\tilde{\boldsymbol{T}}})$ of $G(V, E)$, with vertex set and edge set defined as follows

$$
\begin{aligned}
V_{\tilde{\boldsymbol{T}}} &= \{(u,v) \mid u,v \in V\}, \\
E_{\tilde{\boldsymbol{T}}} &= \{((u,v),(w,x)) \mid u,v,w,x \in V\}.
\end{aligned}
\tag{A.7}
$$

The matrix $\tilde{\boldsymbol{T}}$ indexes each entry using two ordered vertex pairs. The eigenvalues of $\tilde{\boldsymbol{T}}$ are the same as the eigenvalues of $\boldsymbol{T}$ but with the addition of $|V|^2 - 2|E|$ zero eigenvalues due to the additional zeros in the matrix $\tilde{\boldsymbol{T}}$.

Theorem 3. *Let $G(V,E)$ be a k-regular graph with n vertices. Let $\boldsymbol{T}$ be the Perron-Frobenius operator appearing in the Ihara zeta function of $G(V,E)$. The eigenvalues of $\boldsymbol{T}$ are determined by the eigenvalues of the adjacency matrix $\boldsymbol{A}$ of $G(V,E)$.*

Proof. Let λ_A be an eigenvalue of $\boldsymbol{A}$ with corresponding eigenvector $\boldsymbol{\psi} = [\psi_{u_1}, \psi_{u_2}, \ldots, \psi_{u_{|V|}}]'$ where $u_i \in V$, $1 \leq i \leq |V|$. Let λ be an eigenvalue of $\tilde{\boldsymbol{T}}$ with corresponding eigenvector $\boldsymbol{\phi} = [\phi_{(u_1,v_1)}, \phi_{(u_2,v_2)}, \ldots, \phi_{(u_{|V|},v_{|V|})},]'$ where $u_i, v_i \in V, 1 \leq i \leq |V|$. We then have

$$
\lambda\phi_{(u,v)} = \sum_{w,x \in V} \tilde{T}_{(u,v),(w,x)} \phi_{(w,x)}. \tag{A.8}
$$

Substituting the relation in (A.6) into (A.8) and making use of the symmetry of $\boldsymbol{A}$, we have

$$
\begin{aligned}
\lambda\phi_{(u,v)} &= \sum_{w,x \in V} A_{uv} A_{wx} \delta_{vw} \Big(1 - \delta_{ux}\Big) \phi_{(w,x)} \\
&= \sum_{x \in V} A_{uv} A_{vx} \phi_{(v,x)} - A_{uv}\phi_{(v,u)}.
\end{aligned}
\tag{A.9}
$$

We define the vector $\boldsymbol{\phi}$ with entries

$$
\phi_{(v,x)} = A_{vx}\Big(\psi_x - f\psi_v\Big), \tag{A.10}
$$

and likewise

$$
\phi_{(v,u)} = A_{vu}\Big(\psi_u - f\psi_v\Big), \tag{A.11}
$$

$$
\phi_{(u,v)} = A_{uv}\Big(\psi_v - f\psi_u\Big), \tag{A.12}
$$

where f is a complex constant for $G(V,E)$ and $u,v,x \in V$. We now proceed to show that $\boldsymbol{\phi}$ is an eigenvector of $\tilde{\boldsymbol{T}}$. Substituting the relations in (A.10) and (A.11) into the relation in (A.9), we obtain

$$\begin{aligned}
\lambda\phi_{(u,v)} &= \sum_{x\in V} A_{uv}A_{vx}\Big(\psi_x - f\psi_v\Big) - A_{uv}\Big(\psi_u - f\psi_v\Big) \\
&= A_{uv}\Big(\sum_{x\in V} A_{vx}\psi_x - f\psi_v\sum_{x\in V}A_{vx}\Big) \\
&\quad -A_{uv}\psi_u + A_{uv}f\psi_v \\
&= A_{uv}\lambda_A\psi_v - A_{uv}f\psi_v k - A_{uv}\psi_u + A_{uv}f\psi_v \\
&= A_{uv}\Big(\psi_v(\lambda_A - kf + f) - \psi_u\Big) \\
&= \lambda A_{uv}\Big(\psi_v\frac{\lambda_A - kf + f}{\lambda} - \frac{1}{\lambda}\psi_u\Big). \qquad \text{(A.13)}
\end{aligned}$$

From (A.13) we observe that

$$\phi_{(u,v)} = A_{uv}\Big(\psi_v\frac{\lambda_A - kf + f}{\lambda} - \frac{1}{\lambda}\psi_u\Big). \qquad \text{(A.14)}$$

Comparing the relation in (A.14) with our assumption (A.12) we deduce the following equations

$$\frac{\lambda_A - kf + f}{\lambda} = 1, \qquad \text{(A.15)}$$

$$\frac{1}{\lambda} = f. \qquad \text{(A.16)}$$

Solving (A.15) and (A.16) we finally obtain

$$\lambda = \frac{\lambda_A}{2} \pm i\sqrt{k - 1 - \frac{\lambda_A^2}{4}}. \qquad \text{(A.17)}$$

These eigenvalues account for $2|V|$ of the eigenvalues of $\tilde{\boldsymbol{T}}$. The remaining $2|E| - 2|V|$ non-zero eigenvalues take the values ± 1 each with multiplicity $|E| - |V|$. To verify this point we consider the eigenvectors for $\tilde{\boldsymbol{T}}$ of the form

$$\phi_{(u,v)} = A_{uv}w_{uv}. \qquad \text{(A.18)}$$

where w_{uv} is constant corresponding to A_{uv}. Substituting the relation in (A.18) into the relation in (A.9), we have

$$\lambda\phi_{(u,v)} = A_{uv}\sum_{x\in V} A_{vx}w_{vx} - A_{uv}w_{vu}. \tag{A.19}$$

According to the relations in (A.18) and (A.19), $\phi_{(u,v)}$ will an element of the eigenvector for $\tilde{\boldsymbol{T}}$ under the following conditions

$$\begin{aligned}
\sum_{x\in V} A_{ux}w_{ux} &= 0 \ \forall\, u, && \text{(A.20)}\\
w_{uv} &= -w_{vu}, && \text{(A.21)}\\
\text{or} \qquad w_{uv} &= w_{vu}. && \text{(A.22)}
\end{aligned}$$

The values of w_{uv} which correspond to $\boldsymbol{A}$ are irrelevant, so we can set the relations in (A.20) to zero. It is clear that when the relation in (A.21) holds, $\lambda = 1$. On the other hand, when the relation in (A.22) holds, $\lambda = -1$. Additionally, the relation in (A.21) or (A.22) provides us with $|E|$ non-zero variables, and the relation in (A.20) with $|V|$ linear constraints. We thus have $|E| - |V|$ linearly independent solutions for $\lambda = +1$ and $|E| - |V|$ linear independent solutions for $\lambda = -1$, giving $2|E| - 2|V|$ solutions. This completes the spectrum of $\tilde{\boldsymbol{T}}$ and thus $\boldsymbol{T}$.

We can therefore conclude that the spectrum of $\boldsymbol{T}$ is determined by the spectrum of the adjacency matrix $\boldsymbol{A}$ of the graph $G(V, E)$. This completes the proof. □

Theorem 4. *The Ihara zeta function can not distinguish non-isomorphic regular graphs which are cospectral with respect to the adjacency matrix.*

Proof. The Ihara coefficients have a relationship with the spectrum of the Perron-Frobenius operator such that each coefficient can be derived from the elementary symmetric poly-

nomials of the eigenvalue set $\{\lambda_1, \lambda_2, \cdots, \lambda_{2|E|}\}$ of $\boldsymbol{T}$ of the graph $G(V, E)$ as follows

$$\begin{aligned}
c_1 &= -\sum_{k=1}^{2|E|} \lambda_k, \\
c_2 &= \sum_{k=1}^{2|E|} \sum_{p=k+1}^{2|E|} \lambda_k \lambda_p, \\
&\vdots \\
c_r &= (-1)^r \sum_{k_1<k_2< \ \ldots \ <k_r} \lambda_{k_1} \lambda_{k_2} \ \ldots \ \lambda_{k_r}, \\
&\vdots \\
c_{2|E|} &= \prod_{k=1}^{2|E|} \lambda_k.
\end{aligned} \tag{A.23}$$

From Theorem 3, we know that the eigenvalue set $\{\lambda_1, \lambda_2, \cdots, \lambda_{2|E|}\}$ of $\boldsymbol{T}$ of a regular graph $G(V, E)$ is determined by the eigenvalues of the adjacency matrix $\boldsymbol{A}$ of $G(V, E)$. From (A.23), the reciprocal Ihara zeta function is completely determined by the eigenvalue set of $\boldsymbol{T}$. Thus, the regular graphs which are cospectral with respect to the adjacency matrix should also have identical Ihara zeta functions. We can therefore conclude that the Ihara zeta function can not distinguish non-isomorphic regular graphs which are cospectral with respect to the adjacency matrix. This completes the proof. □

A.4 Summary

We have explored the relationship between the Ihara zeta function and discrete-time quantum walks. The analysis hinges around the fact that the Perron-Frobenius operator in the Ihara zeta function can be formulated in terms of the positive support of the transpose of the transition matrix of the discrete-time quantum walks. Furthermore, we have proved that the Ihara zeta function fails to distinguish adjacency cospectral regular graphs. This

supplements Scott and Storm [79] recent research on the Ihara zeta function in distinguishing cospectral graphs. Moreover, it suggests that formulating an alternative zeta function in terms of the characteristic polynomial of $\boldsymbol{T}^n$ $(n \geq 3)$ may solve problems associated with cospectrality.

Bibliography

[1] S. Agarwal, J. Lim, L. Zelnik-Manor, P. Perona, D. Kriegman and S. Belongie. Beyond pairwise clustering. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 838-845, 2005.

[2] S. Agarwal, K. Branson and S. Belongie. Higher-order learning with graphs. In *Proceedings of the International Conference on Machine Learning*, pp. 17-24, 2006.

[3] A. Albarelli, S. Rota Bulo, A. Torsello and M. Pelillo. Matching as a non-cooperative game. In *Proceedings of IEEE Conference on Computer Vision*, pp. 1319-1326, 2009.

[4] X. Bai, E. R. Hancock and R. C. Wilson. Graph characteristics from the heat kernel trace. *Pattern Recognition*, 42(11):2589-2606, 2009.

[5] L. E. Baum and J. A. Eagon. An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology. *Bulletin of the American Mathematical Society*, 73:360-363, 1967.

[6] L. Bartholdi. Counting paths in graphs. *L'Enseignement Mathmatique*, 45:83-131, 1999.

[7] H. Bass. The Ihara-Selberg zeta function of a tree lattice. *International Journal of Mathematics*, 6:717-797, 1992.

[8] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373-1396, 2003.

[9] R. Behmo, N. Paragios and V. Prinet. Graph commute times for image representation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[10] M. Bolla. Spectra, Euclidean representations and clusterings of hypergraphs. *Discrete Mathematics*, 117, 1993.

[11] A. Bretto, H. Cherifi and D. Aboutajdine. Hypergraph imaging: an overview. *Pattern Recognition*, 35(3): 651-658, 2001.

[12] B. P. Brook. The coefficients of the characteristic polynomial in terms of the eigenvalues and the elements of an n×n matrix. *Applied Mathematics Letters*, 19(6):511-515, 2006.

[13] M. Broom, C. Cannings and G. T. Vickers. Multi-player matrix games. *Bulletin of Mathematical Biology*, 59(5):931-952, 1997.

[14] H. Bunke, P. Dickinson, M. Neuhaus and M. Stettler. Matching of hypergraphs — algorithms, applications, and experiments. *Studies in Computational Intelligence*, 91:131-154, 2008.

[15] H. Bunke and K. Shearer. A graph distance metric based on the maximal common subgraph. *Pattern Recognition Letters*, 19(3):255-259, 1998.

[16] T. S. Caetano, J. J. McAuley, L. Cheng, Q. V. Le, A. J. Smola and S. Clara. Learning graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(6):1048-1058, 2009.

[17] P. J. Cameron. Strongly regular graphs. Topics in Algebraic Graph Theory. Cambridge University Press, Cambridge, 203-221, 2004.

[18] G. Chen and G. Lerman. Spectral curvature clustering (SCC). *International Journal of Computer Vision*, 81(3):317-330, 2009.

[19] G. Chen and G. Lerman. Foundations of a multi-way spectral clustering framework for hybrid linear modeling. *Journal of Foundations of Computational Mathematics*, 9:517-558, 2009.

[20] G. Chen, S. Atev and G. Lerman. Kernel spectral curvature clustering (KSCC). In *Proceedings of International Workshop on Dynamical Vision*, pp. 765-772, 2009.

[21] M. Chertok and Y. Keller. Efficient high order matching.*IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2205-2215, 2010.

[22] F. Chung. Spectral graph theory. *American Mathematical Society*, 1992.

[23] F. Chung. The Laplacian of a hypergraph. *AMS DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 10:21-36, 1993.

[24] D. Conte, P. Foggia, C. Sansone and M. Vento. Thirty years of graph matching in pattern recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(3):265-298, 2004.

[25] T. Cour, P. Srinivasan and J. Shi. Balanced graph matching. In *Proceedings of Advances in Neural Information Processing Systems*, pp. 313-320, 2006.

[26] D. Cvetković, P. Rowlinson and S. K. Simić. Eeigenvalue bounds for the signless Laplacian. *Publications de l'Institut Mathmatique*, 81(95):11-27, 2007.

[27] S. I. Daitch, J. A. Kelner and D. A. Spielman. Fitting a graph to vector data. In *Proceedings of International Conference on Machine Learning*, pp. 201-208, 2009.

[28] O. Duchenne, F. R. Bach, I. S. Kweon and J. Ponce. A tensor-based algorithm for high-order graph matching. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1980-1987, 2009.

[29] D. Emms. Analysis of graph structure using quantum walks. Ph.D. Thesis, University of York, 2008.

[30] D. Emms , E. R. Hancock , S. Severini and R. C. Wilson. A matrix representation of graphs and its spectrum as a graph invariant. *Electronic Journal of Combinatorics*, 13(R34), 2006.

[31] D. Emms, S. Severini, R. C. Wilson and E. R. Hancock. Coined quantum walks lift the cospectrality of graphs and trees. *Pattern Recognition*, 42(9):1988-2002, 2009.

[32] M. Ferrer, E. Valveny, F. Serratosa, K. Riesen, and H. Bunke. Generalized median graph computation by means of graph embedding in vector spaces. *Pattern Recognition*, 43(4):1642-1655, 2010.

[33] B. Fischer and J. M. Buhmann. Path-based clustering for grouping of smooth curves and texture segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(4):513-518, 2003.

[34] D. Gibson, J. Kleinberg and P. Raghavan. Clustering categorical data: an approach based on dynamical systems. *VLDB Journal*, 8(4-3):222-236, 2000.

[35] S. Gold and A. Rangarajan. A graduated assignment algorithm for graph matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(4):377-388, 1996.

[36] V. M. Govindu. A tensor decomposition for geometric grouping and segmentation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1150-1157, 2005.

[37] L. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th Annual ACM Symposium on the Theory of Computation*, pp. 212-219, 1996.

[38] L. Hagen and A.B. Kahng. New spectral methods for ratio cut partitioning and clustering. *IEEE Transactions on Computed-Aided Design of Integrated Circuits and Systems*, 11(9):1074-1085, 1992.

[39] C. G. Harris and M. J. Stephens. A combined corner and edge detector. In *Proceedings of Fourth Alvey Vision Conference*, pp. 147-151, 1994.

[40] K. Hashimoto. Artin-type L-functions and the density theorem for prime cycles on finite graphs. *Advanced Studies in Pure Mathematics*, 15:211-280, 1989.

[41] X. He, D. Cai and P. Niyogi. Tensor subspace analysis. In *Proceedings of Advances in Neural Information Processing Systems*, pp. 507-514, 2005.

[42] X. He, D. Cai, H. Liu and J. Han. Image clustering with tensor representation. In *Proceedings of ACM Multimedia*, pp. 132-140, 2005.

[43] Z. He, A. Cichocki, S. Xie and K. Choi. Detecting the number of clusters in n-way probabilistic clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(11):2006-2021, 2010.

[44] Y. Ihara. Discrete subgroups of PL(2, k_φ). In *Proceedings of Symposium on Pure Mathematics*, pp. 272-278, 1965.

[45] Y. Ihara. On discrete subgroups of the two by two projective linear group over p-adic fields. *Journal of the Mathematical Society of Japan*, 18:219-235, 1966.

[46] T. Jebara, J. Wang and S. F. Chang. Graph construction and b-matching for semi-supervised learning. In *Proceedings of International Conference on Machine Learning*, pp. 441-448, 2009.

[47] R. Kondor and K. M. Borgwardt. The skew spectrum of graphs. In *Proceedings of International Conference on Machine Learning*, pp. 496-503, 2008.

[48] R. Kondor, N. Shervashidze and K. M. Borgwardt. The graphlet spectrum. In *Proceedings of International Conference on Machine Learning*, pp. 529-536, 2009.

[49] M. Kotani and T. Sunada. Zeta functions of finite graphs. *Journal of Mathematical Sciences, The University of Tokyo*, 7(1):7-25, 2000.

[50] M. Leordeanu and M. Hebert. A spectral technique for correspondence problems using pairwise constraints. In *Proceedings of IEEE International Conference on Computer Vision*, pp. 1482-1489, 2005.

[51] M. Leordeanu and M. Hebert. Unsupervised learning for graph matching. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 864-871, 2009.

[52] M. Leordeanu and M. Hebert. An integer projected fixed point method for graph matching and MAP inference. In *Proceedings of Advances in Neural Information Processing Systems*, pp. 1114-1122, 2009.

[53] G. Lerman and J. T. Whitehouse. On d-dimensional d-semimetrics and simplex-type inequalities for high-dimensional sine functions. *Journal of Approximation Theory*, 156(1):52-81, 2009.

[54] W. Li and P. Sole. Spectra of regular graphs and hypergraphs and orthogonal polynomials. *European Journal of Combinatorics*, 17:461-477, 1996.

[55] X. Liu, S. Yan, H. Jin. Projective nonnegative graph embedding. *IEEE Transactions on Image Processing*, 19(5):1126-1137, 2010.

[56] D. G. Luenberger. Linear and Nonlinear Programming. Addison-Wesley, 1984.

[57] B. Luo and E. R. Hancock. Structural graph matching using the EM algorithm and singular value decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(10):1120-1136, 2001.

[58] B. Luo, R. C. Wilson and E. R. Hancock. Spectral Embedding of Graphs. *Pattern Recognition*, 36(10):2213-2223, 2003.

[59] A. Mantrach, L. Yen, J. Callut, K. Francoisse, M. Shimbo and M. Saerens. The sum-over-paths covariance kernel: a novel covariance measure between nodes of a directed graph. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 32(6):1112-1126, 2010.

[60] M. Maier, U. von Luxburg and M. Hein. Influence of graph construction on graph-based clustering measures. In *Proceedings of Advances in Neural Information Processing Systems*, pp. 1025-1032, 2008.

[61] H. Mizuno and I. Sato. Bartholdi zeta function of graph coverings. *Journal of Combinatorial Theory, Series B*, 89(1):27-41, 2003.

[62] S. A. Nene, S. K. Nayar and H. Murase. Columbia Object Image Library (COIL-20). *Technical Report CUCS-005-96*, February 1996.

[63] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145-175, 2001.

[64] M. Pavan and M. Pelillo. Dominant sets and pairwise clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):167-172, 2007.

[65] H. Qiu and E. R. Hancock. Clustering and embedding using commute times. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(11):1873-1890, 2007.

[66] J. Ramon and T. Gartner. Expressivity versus efficiency of graph kernels. In *Proceedings of First International Workshop on Mining Graphs, Trees and Sequences*, pp. 65-74, 2003.

[67] P. Ren, R.C. Wilson and E.R. Hancock. Spectral embedding of feature hypergraphs. In *Proceedings of Joint IAPR International Workshop on Structural, Syntactic, and Statistical Pattern Recognition*, pp. 308-317, 2008.

[68] K. Riesen and H. Bunke. Approximate graph edit distance computation by means of bipartite graph matching. *Image and Vision Computing*, 27(7):950-959, 2009.

[69] K. Riesen and H. Bunke. Graph classifcation by means of Lipschitz embedding. *IEEE Transanctions on Systems Man, and Cybernetics, Part B: Cybernetics*, 39(6):1472-1483, 2009.

[70] A. Robles-Kelly and E. R. Edwin. A probabilistic spectral framework for grouping and segmentation. *Pattern Recognition*, 37(7):1387-1405, 2004.

[71] J.A. Rodriguez. On the Laplacian eigenvalues and metric parameters of hypergraphs. *Linear and Multilinear Algebra*, 51:285-297, 2003.

[72] S. Rota Bulo, A. Albarelli, M. Pelillo and A. Torsello. A hypergraph-based approach to affine parameters estimation. In *Proceedings of the International Conference on Pattern Recognition*, pp. 1-4, 2008.

[73] S. Rota Bulo and M. Pelillo. A game-theoretic approach to hypergraph clustering. In *Proceedings of Advances in Neural Information Processing Systems*, pp. 1571-1579, 2009.

[74] S. Rota Bulo, A. Torsello and M. Pelillo. A gameCtheoretic approach to partial clique enumeration. *Image and Vision Computing*, 27(7):911-922, 2009.

[75] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323-2326, 2000.

[76] A. Sanfeliu and K. S. Fu. A distance measure between attributed relational graphs for pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 13(3):353-362, 1983.

[77] I. Sato. A new Bartholdi zeta function of a graph. *International Journal of Algebra and Computation*, 1(6):269-281, 2007.

[78] S. V. Savchenko. The zeta function and Gibbs measures. *Russian Mathematical Surveys*, 48(1):189-190, 1993.

[79] G. Scott and C.K. Storm. The coefficients of the Ihara zeta function. *Involve - A Journal of Mathematics*, 1(2):217-233, 2008.

[80] K. Sengupta and K. L. Boyer. Organizing large structural modelbases. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(4):321-332, 1995.

[81] R. Shankar. Principles of Quantum Mechanics. Second Edition, Plenum, 1994.

[82] L. S. Shapiro and J. M. Brady. Feature-based correspondence: an eigenvector approach. *Image and vision computing*, 10(5):283-288, 1992.

[83] A. Shashua and A. Levin. Linear image coding for regression and classification using the tensor-rank principle. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 623-630, 2001.

[84] A. Shashua, R. Zass and T. Hazan. Multi-way clustering using super-symmetric non-negative tensor factorization. In *Proceedings of the European Conference on Computer Vision*, pp. 595-608, 2006.

[85] B. Shaw and T. Jebara. Structure preserving embedding. In *Proceedings of International Conference on Machine Learning*, pp. 937-944, 2009.

[86] N. Shervashidze and K. M. Borgwardt. Fast subtree kernels on graphs. In *Proceedings of Advances in Neural Information Processing Systems*, pp. 1660-1668, 2009.

[87] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888-905, 2000.

[88] H. M. Stark and A. A. Terras. Zeta functions of finite graphs and coverings. *Advances in Mathematics*, 121:124-165, 1996.

[89] H. M. Stark and A. A. Terras. Zeta functions of finite graphs and coverings, II. *Advances in Mathematics*, 154:132-195, 2000.

[90] H. M. Stark and A. A. Terras. Zeta functions of finite graphs and coverings, III. *Advances in Mathematics*, 208(2):467-489, 2007.

[91] C.K. Storm. The zeta function of a hypergraph. *Electronic Journal of Combinatorics*, 13, 2006.

[92] J. B. Tenenbaum, V. de Silva and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319-2323, 2000.

[93] A. Torsello and A. Robles-Kelly and E. R. Hancock. Discovering shape classes using tree edit distance and pairwise clustering. *International Journal of Computer Vision*, 72(3):259-285, 2007.

[94] W. H. Tsai and K. S. Fu. Subgraph error-correcting isomorphism for syntactic pattern recognition. *IEEE Transactions on Systems, Man, and Cybernetics*, 13(1):48-62, 1983.

[95] S. Umeyama. An eigendecomposition approach to weighted graph matching problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(5):695-703, 1988.

[96] M. A. O. Vasilescu and D. Terzopoulos. Multilinear analysis of image ensembles: tensorFaces. In *Proceedings of the European Conference on Computer Vision*, pp. 447-460, 2002.

[97] S. V. N. Vishwanathan, K. M. Borgwardt, I. R. Kondor and N. N. Schraudolph. Graph kernels. *Journal of Machine Learning Research*, 11:1201-1242, 2010.

[98] C. Wang, Z. Song, S. Yan, L. Zhang and H. J. Zhang. Multiplicative nonnegative greph embedding. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 389-396, 2009.

[99] Y. Watanabe and K. Fukumizu. Graph zeta function in the Bethe free energy and loopy belief propagation. In *Proceedings Neural Information Processing Systems*, pp. 2017-2025, 2009.

[100] Y. Weiss. Segmentation using eigenvectors: A unifying view. In *Proceedings of International Conference on Computer Vision*, pp. 975-982, 1999.

[101] R. C. Wilson and E. R. Hancock. Structural matching by discrete relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(6):634-648, 1997.

[102] R. C. Wilson, E. R. Hancock and B. Luo. Pattern vectors from algebraic graph theory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1112-1124, 2005.

[103] R. C. Wilson and P. Zhu. A Study of Graph Spectra for Comparing Graphs and Trees. *Pattern Recognition*, 41(9):2833-2841, 2008.

[104] A. K. C. Wong, S. W. Lu and M. Rioux. Recognition and shape synthesis of 3D objects based on attributed hypergraphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(3):279-290, 1989.

[105] S. Yan, D. Xu, B. Zhang, H. Zhang, Q. Yang, and S. Lin. Graph embedding and extension: A general framework for dimensionality reduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):40-51, 2007.

[106] S. Yan, D. Xu, Q. Yang, L. Zhang, X. Tang, H.J. Zhang. Discriminant analysis with tensor representation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 526-532, 2005.

[107] J. Yang, S. Yan, Y. Fu, X. Li and T. S. Huang. Non-negative graph embedding. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[108] M. Zaslavskiy, F. Bach and J.-P. Vert. A path following algorithm for the graph matching problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(12):2227-2242, 2009.

[109] R. Zass and A. Shashua. A unifying approach to hard and probabilistic clustering. In *Proceedings of International Conference on Computer Vision*, pp. 294-301, 2005.

[110] R. Zass and A. Shashua. Probabilistic graph and hypergraph matching. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[111] D. Zhao and X. Tang. Cyclizing Clusters via Zeta Function of a Graph. In *Proceedings of Advances in Neural Information Processing Systems*, pp. 1953-1960, 2008.

[112] D. Zhou, J. Huang and B. Scholkopf. Learning with hypergraphs: clustering, classification, and embedding. In *Proceedings of Advances in Neural Information Processing Systems*, pp. 1601-1608, 2007.